

Copyright
by
Andrew Patrick Sharp
2019

The Dissertation Committee for Andrew Patrick Sharp
certifies that this is the approved version of the following dissertation:

**Virtual Fixture Generation for Task Planning with Complex
Geometries**

Committee:

Sheldon Landsberger, Supervisor

Mitchell W. Pryor, Co-Supervisor

Brian O'Neil

Ashish Deshpande

Richard H. Crawford

**Virtual Fixture Generation for Task Planning with Complex
Geometries**

by

Andrew Patrick Sharp

DISSERTATION

Presented to the Faculty of the Graduate School of

The University of Texas at Austin

in Partial Fulfillment

of the Requirements

for the Degree of

DOCTOR OF PHILOSOPHY

THE UNIVERSITY OF TEXAS AT AUSTIN

May 2019

Dedication

To my family: Cyndy, Rick, and Nick

Acknowledgments

I want to thank my parents for their support. I'd also like to thank my advisors Dr. Mitch Pryor and Dr. Sheldon Landsberger for their advice and assistance. Thanks to the other members of my committee Dr. Ashish Deshpande, Dr. Richard Crawford, and especially Dr. Brian O'Neil whose advice while I was at Los Alamos National Laboratory was invaluable. Lastly, I'd like to thank the PT division at Los Alamos National Laboratory.

Abstract

Virtual Fixture Generation for Task Planning with Complex Geometries

Andrew Patrick Sharp, Ph.D.

The University of Texas at Austin, 2019

Supervisors: Sheldon Landsberger

Co-Supervisor: Mitchell W. Pryor

Decontaminating and decommissioning aging nuclear facilities and managing nuclear waste required increased automation to reduce personnel dose. Semi-autonomous behaviors, such as virtual fixtures, aid task execution by managing low-level system resources while operators retain high-level control. Virtual fixtures provide operators with geometric constraints or guidance forces in a robotic manipulator’s workspace. This dissertation advances virtual fixture generation through, the previously unexplored, construction of layers of point cloud based Guidance Virtual Fixtures. Point clouds are used for virtual fixture generation, based on complex surface geometry, to provide more expressive, and therefore useful, environmental representations. Thus, this work builds upon previous point cloud based Forbidden Region Virtual Fixtures to address virtual fixture generation shortcomings outlined in the literature. Task input polygonal mesh checks warn operators if defects are found. Task normal vectors and task parameters are used to calculate point cloud layers at task defined distances from the surface. These layers are interpolated and voxelized to maintain point cloud resolution

at increasing distances from the task surface. The layers are combined into a bi-directional graph structure for storage and future use. The graph structure is combined with a Forbidden Region Virtual Fixture to create a Task Virtual Fixture.

Task Virtual Fixture generation was evaluated with multiple input types including parametric surfaces, polygonal meshes, and point cloud data. Results demonstrate surface model concavity affects the growth in the number of offset layer vertices as does distance from the task surface. Task Virtual Fixture generation intuitively modifies VF layer resolution at extended task surface distances. Point cloud sensor data demonstrated sensor data input for “open world” scenarios. Two visualization and task execution environments were developed to apply Task Virtual Fixtures to spatially discrete and spatially continuous non-contact tasks. The first interface, spatially discrete, was constructed with the Robot Operating System, *RViz*, and *MoveIt!*. This interface displays reachability information to the operator and is called the Manipulator to Task Transform Tool. The second interface allows operators to employ Task Virtual Fixture information in ABB’s RobotStudio for spatially continuous tasks. A small user study was conducted for each interface to demonstrate more expressive Task Virtual Fixtures are still operator interpretable and assist with task execution.

Contents

Acknowledgments	v
Abstract	vi
Contents	viii
List of Tables	xiv
List of Figures	xvi
Glossary	xxx
Acronyms	xxxi
Chapter 1. Introduction	1
1.1 Problem Statement	4
1.2 Review of Relevant Topics	6
1.2.1 Forward and Inverse Kinematics	7
1.2.2 Graph Structures	7
1.2.3 Teleoperation	9

1.2.4	Virtual Fixtures	10
1.2.5	Robot Operating System	11
1.2.6	<i>MoveIt!</i>	13
1.2.7	<i>RViz</i>	15
1.3	Objectives	16
1.4	Organization of Dissertation	17
Chapter 2.	Literature Review	19
2.1	Robots with Semi-autonomous Behaviors at Nuclear Facilities	19
2.2	Additional Robots with Levels of Autonomy and Evaluation Criteria	26
2.3	Virtual Fixtures	27
2.4	Literature Review Summary	35
Chapter 3.	Preliminary Task Virtual Fixture Development	37
3.1	Volumetric Primitive Task Virtual Fixtures	41
3.2	Volumetric Primitive Implementation and Evaluation	44
3.2.1	Discrete Control for Non-Contact Task Execution	47
3.2.2	Continuous Control for Non-Contact Task Execution	54
3.3	Volumetric Primitive Limitations	64

Chapter 4. Complex Geometry Task Virtual Fixture Development	66
4.1 Task Surface Pipeline Input	67
4.2 Polygonal Mesh Verification and Interpolation	72
4.3 Surface Normal Calculations	77
4.4 Virtual Fixture Layer Construction	82
4.5 Guidance Virtual Fixture Data Storage	89
4.6 Summary of Combination Guidance and Forbidden Region Virtual Fixtures .	91
 Chapter 5. Task Virtual Fixture Construction Implementation	 92
5.1 Task Surface Virtual Fixture Software Pipeline	93
5.1.1 Code Optimization	97
5.2 Task Virtual Fixture Visualization	98
5.2.1 Robot Operating System Visualization and Control Interface	98
5.2.1.1 Task Virtual Fixture Discrete Control Interface	102
5.2.1.2 Task Virtual Fixture Continuous Control Interface	105
5.2.2 ABB's RobotStudio Visualization and Control Interface	107
5.3 Task Virtual Fixture Implementation Conclusions	110
 Chapter 6. Task Virtual Fixture Evaluation	 112
6.1 Task Virtual Fixture Evaluation for General Surfaces	112
6.1.1 Parametric Surface Task Virtual Fixture Generation	113

6.1.2	Polygonal Mesh Task Virtual Fixture Generation	118
6.1.3	Point Cloud Task Virtual Fixture Generation	125
6.1.4	Summary of Task Virtual Fixture Generation for General Surfaces . .	127
6.2	Operator Task Virtual Fixture Evaluation	128
6.2.1	Discretized Task Virtual Fixture Operator Evaluation	129
6.2.1.1	Task Surface Test Set Selection	130
6.2.1.2	Experimental Procedure	131
6.2.1.3	Operator Evaluation Results	133
6.2.1.4	Summary of Spatially Discrete Task Operator Evaluation . . .	136
6.2.2	Continuous Task Virtual Fixture Operator Evaluation	137
6.2.2.1	Task Description and Surface Material Selection	138
6.2.2.2	Task Surface Material and Geometry Selection	139
6.2.2.3	Manipulator Hardware and Operator Selection	140
6.2.2.4	Experimental Setup	142
6.2.2.5	Quantitative Evaluation Results	147
6.2.2.6	Qualitative Evaluation Results	152
6.2.2.7	Summary of Spatially Continuous Task Operator Evaluation .	156
6.3	Summary of Task Virtual Fixture Evaluations	157

Chapter 7. Conclusions and Future Extensions	159
7.1 Summary of Task Virtual Fixture Development	160
7.2 Avenues for Future Development	162
7.3 Summary and Significance of Proposed Research	164
Appendices	166
Appendix A. Mobile Manipulator Base Location Variable Normal Surface Virtual Fixture Heat Maps	166
Appendix B. Surface Normals with Varying k Nearest Neighbor Values	169
Appendix C. Crowd Sourced Polygonal Meshes	173
Appendix D. Institutional Review Board Documents	186
Appendix E. Spatially Discrete Operator Evaluation Documents	195
Appendix F. Results from Spatially Discrete TVF Evaluation	198
Appendix G. Spatially Continuous Operator Evaluation Documents	204
Appendix H. Results from Spatially Continuous TVF Evaluation	206
Index	209

Bibliography	210
Vita	224

List of Tables

2.1	Layered Autonomy Modes for Mobile Platforms	22
5.1	Manipulator to Task Transform Tool Menu Options	100
6.1	Average interlayer distances averaged over all nine superellipsoid and supertoroids.	118
6.2	Thingiverse models. Models are available in Appendix C and can be accessed through: " <a href="https://www.thingiverse.com/thing:<model_number>">https://www.thingiverse.com/thing:<model_number> ".	120
6.3	Average interlayer distances percentage of d_{inter} averaged over all nine superel- lipoid and supertoroids.	125
6.4	Spatially discrete Likert scale questionnaire	133
6.5	Test Subject Robotic Manipulator Experience Levels	133
6.6	Spatially continuous Likert scale questionnaire	154
B.1	Normal Angles for Points in Figure 4.9 using Least Square Method (Equation 4.4) with $k = 3$	169
B.2	Normal Angles for Points in Figure 4.9 using Least Square Method (Equation 4.4) with $k = 5$	170
B.3	Normal Angles for Points in Figure 4.9 using Least Square Method (Equation 4.4) with $k = 7$	171

B.4	Normal Angles for Points in Figure 4.9 using Least Square Method (Equation 4.4) with $k = 11$	172
F.1	User Results from Spatially Discrete TVF Evaluation	199
F.2	User Results from Spatially Discrete TVF Evaluation	200
F.3	Average Results from Spatially Discrete TVF Evaluation	201
F.4	Average Results from Spatially Discrete TVF Evaluation	201
F.5	Average Per Trial Results from Spatially Discrete TVF Evaluation	202
F.6	Average Results from Spatially Discrete TVF Evaluation	203
H.1	Results from Spatially Continuous TVF Evaluation	207
H.2	Average Results from Spatially Continuous TVF Evaluation	207
H.3	Likert Survey Results from Spatially Continuous TVF Evaluation	208

List of Figures

1.1	The ARIES Disassembly Module prior to the installation of the end cap (left) [Turner et al., 2009]. Cold Testing of a sphere cleaning robotic system (right) [Turner et al., 2009].	3
1.2	Laser scabbling demonstration selectively removing a 1 x 1m section of concrete in single pass [Khan and Hilton, 2013].	4
1.3	An example of the angles required for a cylindrical task [Sharp and Pryor, 2016].	5
1.4	Bi-directional graph structure where the set of vertices is V and the set of edges is E (Equation 1.1). TVF task parameters are labeled and vertices A, B, C, D are in the same GVF layer, Sharp and Pryor [Sharp and Pryor, 2018].	8
1.5	Telerobotic system overview (left) and different telerobotic control methodologies (right) [Siciliano and Khatib, 2008]	9
1.6	Example illustrations of the VF representations. (a) Point (b) Linear (c) Parametric curve (d) Planar (e) Parametric surface (f) Polygonal mesh (g) Point cloud (h) Volumetric primitive (i) Explicitly described. [Bowyer et al., 2014]	11
1.7	Task surface geometry based VF generation pipeline.	18
2.1	ARIES-DOE stowed (left) and navigating through a three foot aisle [Byrd, 1996].	21

2.2	INL's early iRobot Robotic Platform used for radiation inspection and mapping (left) customized interface used in early efforts at INL to autonomously inspect contaminated environments (right) [Bruemmer et al., 2002].	23
2.3	Example illustrations of the VF representations. (a) Point (b) Linear (c) Parametric curve (d) Planar (e) Parametric surface (f) Polygonal mesh (g) Point cloud (h) Volumetric primitive (i) Explicitly described. [Bowyer et al., 2014]	29
2.4	Argonne's VF. The saw is constrained to be on the plane outside the pipe with arrows in the figure representing allowed motions [DeJong et al., 2006]. . . .	32
2.5	LaserSnake demonstration [Khan and Hilton, 2013].	34
3.1	Cylindrical inspection of an airplane fuselage (left) and inspection locations for the complex surface of an airplane wing (right) in <i>RViz</i>	38
3.2	An example of camera positions around a spherical object in <i>RViz</i>	39
3.3	Visualization of a cylindrical (height = TS_h , radius = TS_r) volumetric primitive task surface (gray), its normals (red arrows), an offset VF surface at $d_{min} = 2 * TS_r$ (blue), and an offset surface at $d_{max} = TS_r$ (green) which is separated from d_{min} by $d_{inter} = TS_r$. Offset VF surface normals (black arrows) face back towards the task surface.	42
3.4	Visualization of a volumetric primitive VF implementation. VF layers (blue and green) and their surface normals (black arrows) are only calculated for the task surface (gray cylinder, height = TS_h and radius = TS_r) region of interest.	45

3.5	A VF layer (red markers) within a VNSVF around a cylindrical volumetric primitive FRVF (green cylinder) representing a storage canister. The <i>RViz</i> visualization also displays an ASUS Xtion Pro Live RGBD camera on a UR5 (lower right). Motion Command <i>RViz</i> plugin (bottom left) for TVF navigation [Sharp and Pryor, 2016].	46
3.6	The NRG VaultBot mobile manipulator with attached Robotiq two finger gripper and RealSense R200 RGBD camera (left) and the VaultBot loaded into <i>RViz</i> with <i>MoveIt!</i> running. The orange version of the arms are the goal positions and can be altered with the 6 DOF red, green, and blue interactive markers (right).	49
3.7	The original storage can used for testing (left) and the Savy 4000 container being implemented at LANL (middle). Virtual workspace constructed from environmental information based on the lab setup at UT Austin (right). . . .	50
3.8	VaultBot visualization in <i>RViz</i> with environmental information (volumetric primitive, shelf, floor plane) and locations above the reachability percentage threshold marked in blue squares (left). VaultBot workspace (right) for the left UR5 (red) and right UR5 (blue).	50
3.9	Mobile base placement for task execution results at 40 cm displayed as a heat map scaled in meters (right).	51
3.10	The NRG VaultBot mobile dual manipulator <i>RViz</i> environment with shelf and object collision meshes. RGBD pointcloud data and Motion Command <i>RViz</i> plugin (bottom left) for TVF navigation are overlaid as the operator would see. 53	

3.11 Overview of a pipe inspection task (left) and GVF maintained operator view-point for task execution (right).	54
3.12 Motion Command <i>RViz</i> plugin for VNSVF navigation (left). SIA20 with a laser cutter tool in the <i>RViz</i> environment with 6 degrees of freedom interactive marker control (right).	56
3.13 Process schematics and complete cutting head assembly [Khan and Hilton, 2013].	57
3.14 Demonstration showing planar shape primitive VNSVF poses (red markers) with 20 cm (left) and 10 cm (right) spacing and a planned path (yellow lines).	58
3.15 Demonstration showing planar shape primitive VNSVF poses (red markers) with 10 cm spacing and a planned multiple-pass ‘gouge’ cut (yellow lines) utilizing several GVF layers.	59
3.16 Demonstration showing cylindrical shape primitive VNSVF poses (red markers) with 10 cm spacing and a planned path (yellow lines).	60
3.17 “Nuclear Jungle” tube cutting demonstration before (left) and after (right) from [Khan and Hilton, 2013].	61
3.18 Demonstration of the “nuclear jungle” with cylindrical shape primitive VNSVF poses (red markers) with 10 cm spacing around 60 mm piping and the planned paths (yellow lines).	62
3.19 Demonstration of the “nuclear jungle” with planar shape primitive VNSVF poses (red markers) with 10 cm spacing with the planned path avoiding piping (yellow lines).	63

4.1	TVF generation algorithms to go from a polygonal mesh or sensor data to a bi-directional graph structure.	67
4.2	Spline based surface (left), low detail STL (middle), and high detail STL (right) [PTC, 2017].	68
4.3	Filling in polygonal mesh holes (left), placing control points, orange points and lines (middle), and conversion to NURBS patches, blue and yellow lines (right) in PolyWorks Modeler [Innvometric, 2018b].	70
4.4	Point cloud data merging (1-5 into 6) and conversion into a polygonal mesh (7) in PolyWorks Inspector [Innvometric, 2018a].	71
4.5	FARO Cobalt 9MP (left) and Faro Cobalt mounted to a Universal Robotics UR3 on a granite measurement table (right).	71
4.6	Screenshots of the searches of ‘table’ (left) and ‘chair’ (right) of Google 3D warehouse [SketchUp, 2017a].	72
4.7	Box (left) and bowl (right) <i>SketchUp</i> [SketchUp, 2017b] models.	73
4.8	Visualization of the triangle interpolation process from Algorithm 4. Original points and sides are in black. Added points and sides are blue.	76
4.9	Example of the effect of variation of k nearest neighbors for normal vector calculations on small features. The values of k increase from bottom to top (3, 5, 7, 11). Numerical values were calculated through Least Squares Method (Equation 4.4) and are presented in Appendix B.	79

4.10	Superellipsoid example of $k_{nearest}$ neighbor (5 top left, 20 top right) and r_{radius} neighborhood ($r_{radius} = d_{intra}$ middle left, $r_{radius} = 5 * d_{intra}$ middle right) surface normal estimation. Example of d_{intra} effects on STL interpolation and surface normal calculation (0.5 m) bottom left, (0.5 m) bottom right). Viewed with PCLVisualizer [PCL, 2018].	81
4.11	Visualization of a hemispherical task surface and VF normal vectors (Equation 4.5). Two VF layers are displayed on the convex outside and three VF layers on the concave inside. The number of vectors decreases with distance on the concave interior.	83
4.12	Visualization of a point cloud (red and green points), the current pose (black), the r_{radius} distance, the lower limit distance, the points included in interpolation (green), the points not included in interpolation (red), and the Algorithm 6 interpolated points (blue).	84
4.13	The first three VF layers in an interior slot.	86
4.14	The sixth VF layer demonstrating how interior spaces can result in complex point clouds which are not conducive to conversion into a mesh.	87
4.15	Three VF layers around a superellipsoid were converted into polygonal meshes to function as FRVFs. Half the FRVFs have been removed to visualize interior surfaces (left). A closeup of the superellipsoid and FRVF surfaces (right). This demonstrates conversion at multiple distances but only one FRVF is required for a task.	88
4.16	The sixth VF layer with spherical forbidden regions added.	89

4.17	Bi-directional graph structure where the set of vertices is V and the set of edges is E (Equation 4.6). TVF task parameters are labeled and vertices A, B, C, D are in the same GVF layer, Sharp and Pryor [Sharp and Pryor, 2018].	90
5.1	TVF generation algorithms to go from a polygonal mesh or sensor data to a bi-directional graph structure.	93
5.2	Surface meshing (left) [Rusu and Cousins, 2017b], Visualization (middle) [Rusu and Cousins, 2017b], Surface normal estimation viewpoint corrected (right) [Rusu and Cousins, 2017a]	95
5.3	TVF visualization and control interface showing current vertex as a white marker and blue markers represent different GVF layers. The marker in the white circle has rotation about its normal altered by the operator between the left and right images.	101
5.4	Display of the three developed TVF graph visualization methods: full GVF layer visualization (left), $k_{nearest} = 10$ neighbors (middle), edge weight decreasing (right). Reachability varies due to changes in task surface location. . . .	102
5.5	TVF vertex reachability analysis visualization with the current vertex (white), different GVF layer vertices (blue), same GVF layer reachable vertices (green), and same GVF layer unreachable vertices (red) (left). Visualization of the task FRVF surface and 6DOF interactive marker control (right).	104
5.6	Full spatially discrete task <i>RViz</i> visualization and control interface showing the task FRVF, task surface interactive marker, TVF vertex reachability analysis, and <i>RViz</i> visualization controls.	105

5.7	Close up visualizations of the displayed path graph (left, middle) and highlighted failure vertex (right, white circle).	106
5.8	Example of TVF poses loaded into ABB's RobotStudio [ABB, 2018b] for operator utilization.	108
6.1	A subset of the VTK generated superellipsoid models. Parameters vary from the top left ($N_z, N_{xy} = 0$) to the bottom right ($N_z, N_{xy} = 4$). No regions with inverted surface normals are present.	114
6.2	A subset of the VTK generated supertoroid models. Parameters vary from the top left ($N_z, N_{xy} = 0$) to the bottom right ($N_z, N_{xy} = 4$). The models also show inverted $\hat{\mathbf{n}}$ regions (dark gray).	115
6.3	Graph of the number of VF layer vertices at distances of 0.05 m, 0.55 m, 1.05 m from the surface for superellipsoids and supertoroids. Supertoroid results are more erratic due to regions with incorrect surface normals.	117
6.4	VF layer resolutions for three VF layers at distances of 0.05 m, 0.55 m, 1.05 m for both superellipsoids and supertoroids. Superellipsoid resolutions are highly similar but supertoroid results are more erratic due to regions with incorrect surface normals.	118
6.5	Intralayer graph edges at distances of 0.05 m, 0.50 m, 1.05 m for superellipsoids and superellipsoids. Supertoroid results are more erratic due to regions with incorrect surface normals.	119

6.6	Interlayer graph edges between 0.05 m, 0.50 m, 1.05 m distances for superellipsoids and superellipsoids. Supertoroid results are more erratic due to regions with incorrect surface normals.	120
6.7	Thingiverse model 3119803's interior holes with incorrect surface normals visualized in MeshLab [MeshLab, 2018]. Light gray surfaces have normals pointing toward the viewpoint while dark gray surfaces have normals pointing away from the viewpoint. The holes in the bracket are dark gray when viewed from outside the surface (left) and surface normals point toward the viewpoint when it is inside the bracket with dark gray interior surfaces above and below (right).	121
6.8	Parametric and polygonal mesh models evaluation results with varying intralayer distances. VF layer sizes were averaged per layer from 10 to 100 times $\bar{T}(i)_{side}$. Results show layer size growth rates of approximately $N = C * dist^2$, where N is the number of points and C is a constant. A comparison line of $0.005 * dist^2$ is displayed.	122
6.9	Parametric and polygonal mesh models evaluation resolution with varying intralayer distances. The r_{res} neighbors were averaged per layer for distances from 10 to 100 times $\bar{T}(i)_{side}$	123
6.10	Thingiverse models wrench (1187995, top left), rocket (3101067, top right), antlers (3108035, bottom left), and icosahedron (3119665, bottom right) each with generated surface points (red dots), first (green dots), and second (blue dots) GVF layer points with point normals represented as white lines.	124

6.11	TVF generation pipeline tested on PCN data taken at UT Austin’s mock of the H-canyon tunnel [Pryor and Landsberger, 2017] (top).	126
6.12	Point cloud data evaluation from 0.05 m to 1.05 m with varied intralayer distances 0.1 m and 0.5 m. VF layer sizes are compared to supersolid testing data.	127
6.13	The results are visualiized using the MTTT. A SIA20 previously used in this research thread for D&D tasks replaced the NRG VaultBot which gathered the data. The interface provides reachability feedback. Red: current GVF layer but unreachable. Green: current GVF layer but reachable. White: current pose in TVF. Blue: pose in layer closer to or farther from the task surface. .	128
6.14	MTTT interface showing interactive markers for task FRVF and SIA20 manipulator control. Pose and reachability feedback is sumarized and displayed for the current task FRVF location. Red: current GVF layer but unreachable. Green: current GVF layer but reachable. White: current pose in TVF. Blue: pose in layer closer to or farther from the task surface.	130
6.15	A subset of the VTK generated superellipsoid models. Parameters vary from the top left ($N_z, N_{xy} = 0$) to the bottom right ($N_z, N_{xy} = 4$). The models also show the surface points (green dots), their surface normal (white lines), and the first TVF layer (blue dots) (left) [Sharp and Pryor, 2018]. Thingiverse Moai monolith model (https://www.thingiverse.com/thing:908062) (right). .	131
6.16	T1 and T2 averaged trial time (left) and reachable percentage (right) for the four trials.	134

6.17	T1 and T2 averaged Likert scale responses (Table 6.4).	135
6.18	PVATePla PlasmaPen Atmospheric Plasma System [PVATePla, 2018].	138
6.19	Complex and asymmetric model of flag waving in the wind.	140
6.20	The Motoman SIA5 (left) and SIA10 (right) 7 DOF manipulators with tooling from previous projects at LANL.	141
6.21	ABB IRB 140 chosen for TVF hardware operator evaluation.	142
6.22	ABB IRB 140 hardware setup for TVF operator evaluation with the elevated workspace platform (bottom) and the test surface raised on machining blocks (bottom center).	143
6.23	Close up of TVF operator evaluation plasma pen and task surface.	145
6.24	MTTT representation of the spatially continuous testing surface with reacha- bility analysis for the Yakasawa SIA20.	146
6.25	RobotStudio interface for spatially continuous testing with the IRB140 (left) and a closeup of the TVF frames (right).	147
6.26	Spatially continuous testing setup time data for all four operators and the average.	148
6.27	Spatially continuous testing execution time data for all four operators and the average.	149
6.28	Spatially continuous plasma testing raw data for manual (left), low resolution TVF (middle), high resolution TVF (right) tests for all four operators. The gap represents and incomplete test.	150

6.29	Spatially continuous plasma testing analyzed data for manual (left), low resolution TVF (middle), high resolution TVF (right) tests for all four operators. The gap represents and incomplete test. Pixel data was binned into unclean (white), clean (gray), and over clean (black).	151
6.30	Spatially continuous testing unclean, clean, and over clean surface percentages with manual, low resolution TVF poses, and high resolution TVF poses. . .	152
6.31	Percentage of surface cleaned per minute of setup time for spatially continuous testing with manual, low resolution TVF poses, and high resolution TVF poses.	153
6.32	Average responses for Likert scale questions one through four for manual assignment, low resolution TVF, and high resolution TVF testing.	155
6.33	Average responses for Likert scale questions five through eight for manual assignment, low resolution TVF, and high resolution TVF testing.	156
A.1	Mobile base placement for task execution results at 20 cm (left) and 25 cm (right) displayed as a heat map scaled in meters.	166
A.2	Mobile base placement for task execution results at 30 cm (left) and 35 cm (right) displayed as a heat map scaled in meters.	167
A.3	Mobile base placement for task execution results at 40 cm (left) and 45 cm (right) displayed as a heat map scaled in meters.	167
A.4	Mobile base placement for task execution results at 50 cm (left) and 55 cm (right) displayed as a heat map scaled in meters.	168
C.1	Thingiverse model 17314, https://www.thingiverse.com/thing:17314	173

C.2	Thingiverse model 38840, https://www.thingiverse.com/thing:38840	174
C.3	Thingiverse model 70549, https://www.thingiverse.com/thing:70549	174
C.4	Thingiverse model 906951, https://www.thingiverse.com/thing:906951	175
C.5	Thingiverse model 908062, https://www.thingiverse.com/thing:908062	175
C.6	Thingiverse model 1014845, https://www.thingiverse.com/thing:1014845	176
C.7	Thingiverse model 1187995, https://www.thingiverse.com/thing:1187995	176
C.8	Thingiverse model 1677784, https://www.thingiverse.com/thing:1677784	177
C.9	Thingiverse model 2120591, https://www.thingiverse.com/thing:2120591	177
C.10	Thingiverse model 3101067, https://www.thingiverse.com/thing:3101067	178
C.11	Thingiverse model 3106129, https://www.thingiverse.com/thing:3106129	178
C.12	Thingiverse model 3108035, https://www.thingiverse.com/thing:3108035	179
C.13	Thingiverse model 3108554, https://www.thingiverse.com/thing:3108554	179
C.14	Thingiverse model 3119494, https://www.thingiverse.com/thing:3119494	180
C.15	Thingiverse model 3110862, https://www.thingiverse.com/thing:3110862	180
C.16	Thingiverse model 3114718, https://www.thingiverse.com/thing:3114718	181
C.17	Thingiverse model 3118241, https://www.thingiverse.com/thing:3118241	181
C.18	Thingiverse model 3118847, https://www.thingiverse.com/thing:3118847	182
C.19	Thingiverse model 3118855, https://www.thingiverse.com/thing:3118855	182
C.20	Thingiverse model 3119665, https://www.thingiverse.com/thing:3119665	183

C.21	Thingiverse model 3119580, https://www.thingiverse.com/thing:3119580	183
C.22	Thingiverse model 3119670, https://www.thingiverse.com/thing:3119670	184
C.23	Thingiverse model 3119735, https://www.thingiverse.com/thing:3119735	184
C.24	Thingiverse model 3119802, https://www.thingiverse.com/thing:3119802	185
C.25	Thingiverse model 3119803, https://www.thingiverse.com/thing:3119803	185
D.1	IRB proposal for TVF operator evaluation page one.	187
D.2	IRB proposal for TVF operator evaluation page two.	188
D.3	IRB site approval letter for TVF operator evaluation.	189
D.4	IRB consent form for TVF operator evaluation page one.	190
D.5	IRB consent form for TVF operator evaluation page two.	191
D.6	IRB approval letter for TVF operator evaluation page one.	192
D.7	IRB approval letter for TVF operator evaluation page two.	193
D.8	IRB approval letter for TVF operator evaluation page three.	194
E.1	Script for spatially discrete task MTTT operator evaluation page one.	196
E.2	Script for spatially discrete task MTTT operator evaluation page two.	197
G.1	Script for spatially continuous task operator evaluation with ABB software and hardware.	205

Glossary

context switching Translating to/from the input, output, and possibly intermediary contexts of the operator/robot interface. 37

Descartes ROS-I package whose goal is to provide a pipeline structure which utilizes planners and filters to transform an under-defined Cartesian plan into a joint path plan. 54, 55, 64, 100, 105–107, 111, 137, 140

Motion Command NRG developed RViz plugin for Virtual Fixture navigation. xviii, xix, 46, 47, 51–56

MoveIt! ROS’s manipulator motion planning and scene management package. vii, xviii, 12–15, 44, 45, 47–49, 55, 92, 103, 111, 161, 164

RViz 3D robot visualization environment paired with ROS. vii, xvii–xix, xxii, 13, 15, 18, 32, 38, 39, 44, 46, 49, 50, 53, 56, 92, 98, 103, 105, 107, 111, 129, 161, 164

SketchUp 3D modelling software from Trimble, <https://www.sketchup.com/>. xx, 72–74

Acronyms

ALARA As Low As Reasonably Achievable. 1

ANL Argonne National Laboratory. 31

ARIES-DOE Autonomous Robotic Inspection Experimental System. xvi, 20, 21

ARIES-LANL Advanced Recover and Integrated Extraction System. 20, 23

BGL Boost Graph Library. 92, 97, 110, 161

D & D Decontaminating and Decommissioning. 2, 19, 31, 47, 55, 64, 129, 131

DAE COLLADA: COLLABorative Design Activity. 72

DOE Department of Energy. 1, 20, 27, 44

DOF Degrees of Freedom. xviii, xxii, xxvi, 7, 14, 15, 48, 49, 104, 140, 141, 162

EEF End Effector. 7, 16, 24, 31–33, 38–40, 44, 45, 48, 52, 53, 66, 100, 103, 107, 108, 126, 129, 142

EM DOE Office of Environmental Management. 25

FK Forward Kinematics. 7, 14

FRVF Forbidden Region Virtual Fixture. xviii, xxi, xxii, xxv, 11, 17, 27–29, 31, 36, 40, 41, 45, 46, 52, 64, 82, 86–88, 91, 96, 98, 103–105, 127, 130, 159–162

GVF Guidance Virtual Fixture. xvi, xix, xxii, xxiv, xxv, 8, 11, 16, 17, 27, 29, 36, 40, 41, 44, 52, 54, 55, 57–59, 64, 66, 86, 89–91, 95–99, 101, 102, 104, 107, 109, 112, 116, 123, 124, 127–130, 146, 159–163

IK Inverse Kinematics. 7, 14, 47, 103, 106

INL Idaho National Laboratory. xvii, 21, 23

IRB Institutional Review Board. xxix, 129, 187–194

LANL Los Alamos National Laboratory. xxvi, 20, 23, 24, 27, 35, 48, 129, 137, 140, 141, 164, 165

LOA Levels of Autonomy. 16, 19, 21, 24–26, 35, 48, 55, 66, 137

MTTT Manipulator to Task Transform Tool. vii, xxv, xxix, 103, 105, 112, 126, 128–131, 135–137, 157, 163–165, 196, 197

NRG Nuclear Robotics Group. xviii, 13, 19, 22, 23, 25, 32, 46–49, 53

OMP Open MP. 92, 110, 161

PCL Point Cloud Library. 92, 94, 95, 110, 161

PCN Point Cloud with Normals. xxv, 85, 86, 91, 93–97, 107, 112, 116, 126, 127, 157, 161, 162

PPE Personal Protective Equipment. 31, 40

PRM Probabilistic Road Map. 14

RC Robot-Centered. 10

RGBD Red-Green-Blue-Depth. xviii, 13, 15, 28, 29, 48, 49, 51, 53, 64

ROS Robot Operating System. vii, 11–15, 44, 46, 48, 54, 64, 92, 98, 99, 105, 107, 110, 111, 140, 160, 161, 164

ROS-I ROS-Industrial. 13, 44, 129

RRT Rapidly-exploring Random Tree. 14

STL STereoLithography. xxi, 35, 81, 92, 93, 98, 110, 116

TC Task-Centered. 10, 45, 46

TVF Task Virtual Fixture. xv, xvi, xviii, xx, xxii, xxiii, xxv–xxvii, xxix, 8, 17, 18, 37, 40, 41, 46, 52, 53, 64–67, 69, 70, 74, 79, 84, 85, 90–93, 95–113, 115, 116, 118, 119, 123, 125–132, 134–137, 141–143, 145–165, 187–194, 199–203, 207, 208

URDF Unified Robot Description Format. 12, 14, 47

VF Virtual Fixture. xvi–xviii, xxi, xxiii–xxv, 10, 11, 17, 18, 27–29, 31, 32, 35, 36, 38–46, 64, 66, 67, 79, 82–91, 95, 97, 112, 116–118, 122, 125, 127, 159–162, 164, 165

VNSVF Variable Normal Surface Virtual Fixture. xviii, xix, 41, 44–49, 51, 53–60, 62–64, 66, 89, 98, 99, 102, 103, 105, 110, 129, 136, 160

VP View Point. 10

Chapter 1

Introduction

For decades, nuclear material production was prioritized over environmental concerns resulting in legacy waste which is poorly documented but must be managed, size-reduced, sorted, and properly disposed. The potentially harmful nature of radioactive waste, including manufacturing equipment, storage components, and outdated laboratory equipment, motivates the use of automation and remote systems to minimize operator dose [United States Nuclear Regulatory Commission, 2018]. The acronym for this pursuit is As Low As Reasonably Achievable (ALARA). It is defined in 10 CFR 20.1003 and means taking all reasonable efforts to minimize personnel exposure to ionizing radiation. ALARA dates back to the early nuclear industry and has driven technological advancements in the nuclear complex for decades [Saling, James and Fentiman, Audeen, 2001]. The mission of the United States Department of Energy (DOE) is:

“to ensure America’s security and prosperity by addressing its energy, environmental and nuclear challenges through transformative science and technology solutions.” [United States Department of Energy, 2015]

As such, the nation’s nuclear weapons program and radioactive waste disposal are two of the DOE’s primary responsibilities. Furthermore, automation is desired to mitigate potential hazards and the DOE is pursuing new automation for national laboratories [Wood,

1996; Turner et al., 2009] and Decontaminating and Decommissioning (D & D) [DOE, 2016] processes. These operations often take place in unstructured or heavily constrained environments such as a glovebox (Figure 1.1). Tasks often require a unique approach and the need to reduce contact between humans and radioactive waste led to the development of several systems. At Argonne National Laboratory a robotic saw was used for a D & D sectioning pipe task [DeJong et al., 2006] in order to remove personnel from the contaminated environment. Other systems include the Advanced Recovery and Integrated Extraction System (ARIES) [Turner et al., 2009] for nuclear manufacturing and decommissioning at Los Alamos National Laboratory (LANL). ARIES advanced nuclear domain robotics and recent upgrades include semi-autonomous behaviors such as pre-programming to retrieve and exchange grippers in the Disassembly Module [Turner et al., 2009]. Also, the Robotic Integrated Packaging System (RIPS) Module which handles material canning is capable of multiple LOAs including autonomous, semi-autonomous, or manual fault recovery. Harden et al. conclude that “...considerable opportunity for improved nuclear material handling and processing capabilities with reduced occupational radiation exposure is made possible by automation technologies” [Turner et al., 2009]. Another LANL robotic system was designed to clean out spherical containment vessels. It uses multiple LOAs including teleoperation, semi-autonomous operation, and automatic modes of operation [Harden and Pittman, 2008]. The controllers have a number of different operating frames including global, spherical, and EEf Cartesian. Other approaches seek to construct a more generalized framework based on force and torque data. This framework is applicable to a variety of tasks, including cabinet door opening, by establishing LOAs such that lower LOAs are accessible should a system fault occur [Schroeder and Pryor, 2011].



Figure 1.1: The ARIES Disassembly Module prior to the installation of the end cap (left) [Turner et al., 2009]. Cold Testing of a sphere cleaning robotic system (right) [Turner et al., 2009].

More recently, the United Kingdom's Sellafield facility has had site lifetime decommissioning costs rise above £67.5 billion [House of Commons Committee of Public Accounts, 2013]. As such, the Sellafield Decommissioning Program has investigated robotic fiber optic lasers for concrete scabbling (Figure 1.2) and size reduction of large cylinders [Khan and Hilton, 2013; Hilton and Khan, 2014]. In [Khan and Hilton, 2013] pipes were cut in a single or double pass with a linear motion. Teleoperation is an alternative but can be costly and time consuming to train operators. Also, tasks may still fail as noted in [DeJong et al., 2006]. Hilton et al. comment:

Future decommissioning of nuclear facilities will make increasing use of non-contact remote cutting techniques... ...What is needed is a highly automated remote technology that can deliver a non contact smarter dismantling process, cut

most materials, cut **complicated structural geometries**... [Hilton and Khan, 2014]

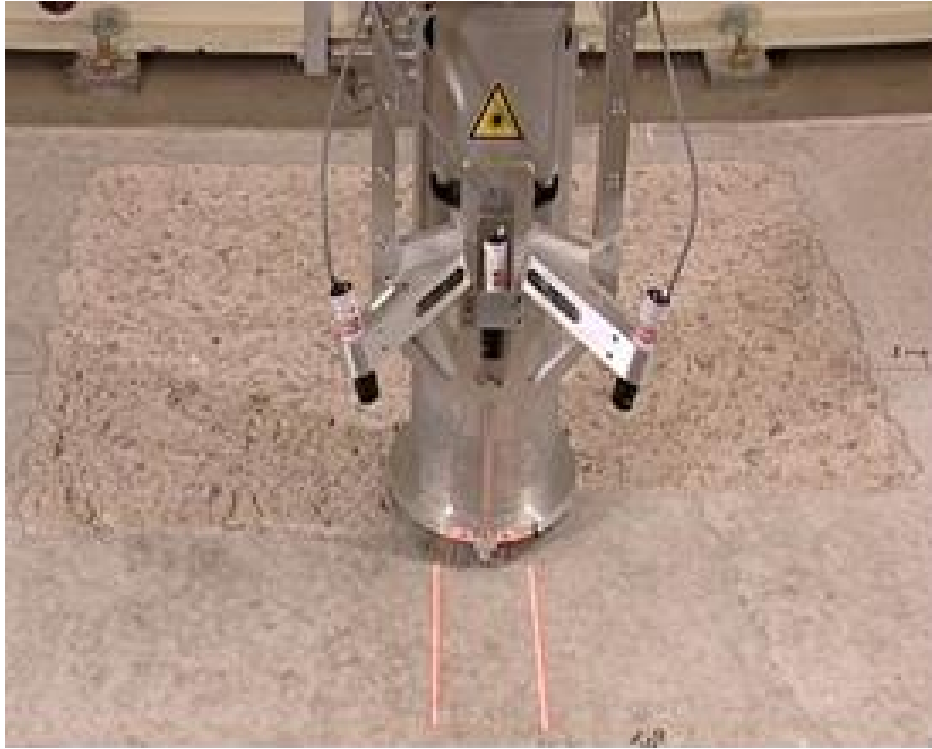


Figure 1.2: Laser scabbling demonstration selectively removing a 1 x 1m section of concrete in single pass [Khan and Hilton, 2013].

1.1 Problem Statement

Ideally, robotic systems will be operated by trained technicians with little or no intervention from robotics experts. However, there are two significant barriers to adopting robotics and remote systems. First is the *correspondence problem* where differences between operator and manipulator kinematics complicate motion planning. Second is *context switching*

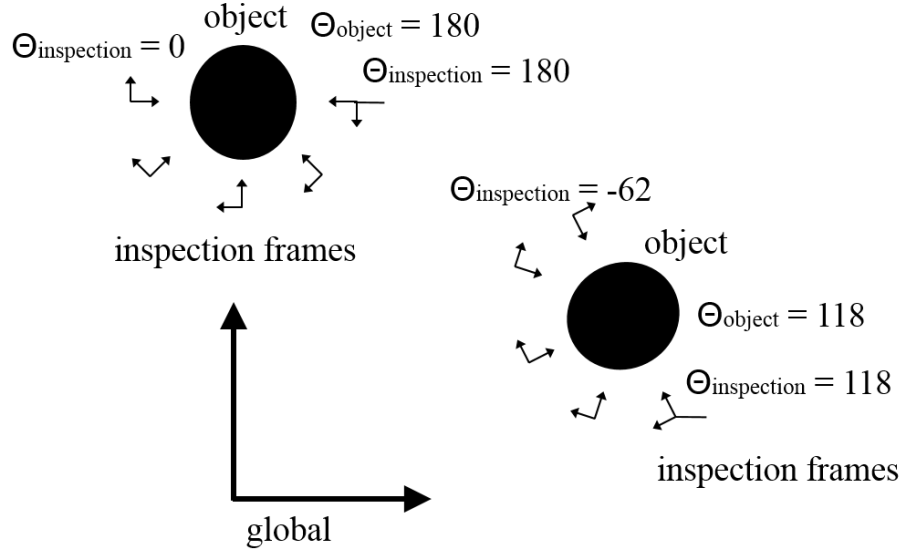


Figure 1.3: An example of the angles required for a cylindrical task [Sharp and Pryor, 2016].

which is translating to/from the operator's viewpoint input, end effector (EEF) frame output, and possibly intermediary frames of the operator/robot interface, which can unnecessarily burden the operator's mental load. For example, placing a tool in the proper location relative to a flat task surface aligned with the global frame is not particularly difficult. The operator must control the tool angle to the object accounting for the transform between the operator & world frame, the world & manipulator frames, and the manipulator & tool frames while also maintaining the sensor offset from the surface. If the task surface is cylindrical with a specified surface offset, such as a storage drum inspection or pipe reduction, the chain of transformations become a significant problem (Fig. 1.3). Task surface complexity exacerbates this issue and increases the chance of error until remote task completion becomes untenable.

Operators who cannot anticipate time-saving and precise semi- or fully autonomous

behaviors lack trust in a robot’s ability to complete a task, and use of system experts as operators is impractical. To limit operational complexity and task duration, low-level system resources are managed by the system with only high-level user input. Operator assistance via semi-autonomous behaviors or artificial EEF motion guidance/constraints has been shown in the literature to reduce operator mental load and increase efficiency. Examples include obstacle avoidance and collision detection in multiple domains studied including mobile manipulation [Hebert et al., 2015; Ciocarlie et al., 2012], surgical [Ren et al., 2008; Li et al., 2007], and nuclear [Turner et al., 2009; House of Commons Committee of Public Accounts, 2013] research areas. Environments studied also include cluttered home environments [Ciocarlie et al., 2012] and human anatomy variability [Li et al., 2007] but nuclear environments are comparatively static. Radioactive waste (which includes original manufacturing facilities, abandoned machinery, and facility piping infrastructure) often comes in complex shapes, most robotic systems lack the flexibility to intelligently adjust to unique pieces or complex piping arrangements [Turner et al., 2009; Khan and Hilton, 2013; Hilton and Khan, 2014].

1.2 Review of Relevant Topics

The solution to this problem lies at the intersection of several topical areas related to robotics which are quickly introduced in this section. These topics include graph structures, serial manipulator Forward & Inverse Kinematics, teleoperation, and virtual fixtures. Readers already familiar with robotics may choose to skip ahead to the objectives in Section 1.3. A review of current literature associated with related solutions is provided in Chapter 2.

1.2.1 Forward and Inverse Kinematics

Forward Kinematics (FK) is the process of finding the EEF pose (position and orientation) in 3D space based on a robotic description and known joint values. Inverse Kinematics (IK) refers to the process of calculating joint values based on a robotic description and desired EEF pose. The robot description is a list of parameters describing the relationship between joint axes. These parameters are based on the work of Jacques Denavit and Richard Hartenberg in the 1950's and as such are referred to as DH parameters [Denavit and Hartenberg, 1955, 1964]. The IK problem increases in difficulty with the number of joints. Once the system has more than 6 DOF, a closed-form solution is no longer possible and numerical methods are required. Multiple IK solutions are possible depending on the number of joints and desired pose. However, multiple solution manifolds in the joint space allow for solution optimization.

1.2.2 Graph Structures

A graph, G , consists of a finite set V of vertices and a finite set E of edges (Equation 1.1). The set of vertices V are connected in pairs which are stored in a set of edges E . The graph in Fig. 1.4 has six vertices A, B, C, D, E, F , and eight edges. Vertices connected by an edge, e , are called the end-vertices of e . Vertices A and B are the end vertices of edge (A, B) and thus are adjacent. Vertex B has the neighbors A, C , and D . Edges contain the weighting values between vertices. Connections between vertices can be directional, from vertex A to vertex B , or bi-directional. An adjacency list or an adjacency matrix is generally used to store edge information. Graphs have applications in many fields of science and engineering including motion planning, robotics, and spatial trees [Rahman, 2017].

$$G = (V, E)$$

$$V = \{A, B, C, D, E, F\} \quad (1.1)$$

$$E = \{AB, AC, AD, AE, AF, BC, BD, CD\}$$

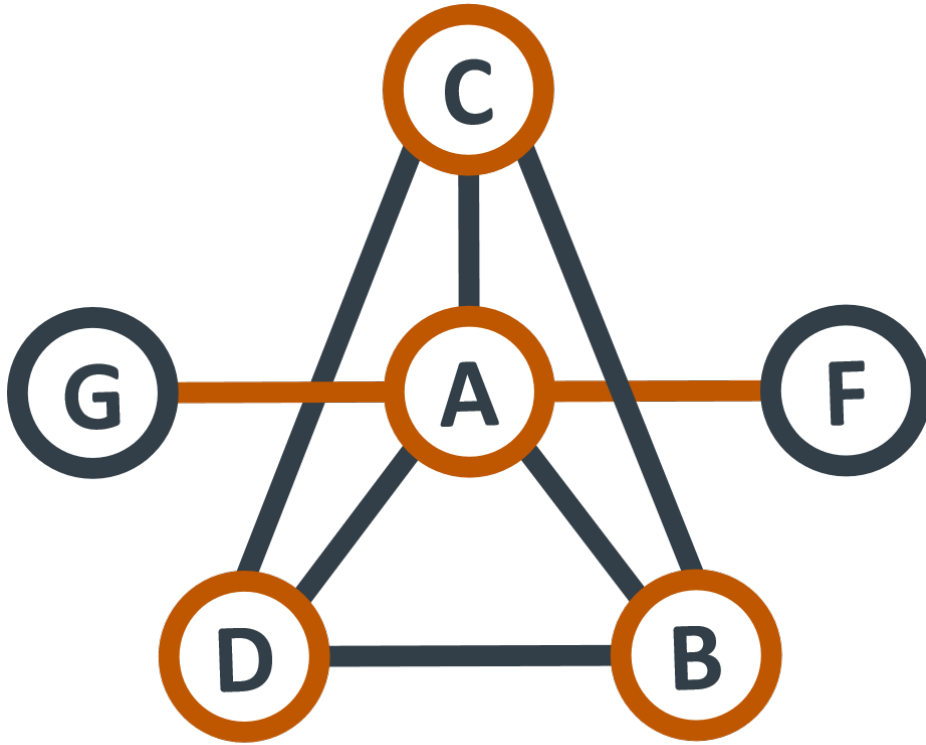


Figure 1.4: Bi-directional graph structure where the set of vertices is V and the set of edges is E (Equation 1.1). TVF task parameters are labeled and vertices A, B, C, D are in the same GVF layer, Sharp and Pryor [Sharp and Pryor, 2018].

1.2.3 Teleoperation

Teleoperation generally implies there is a barrier between the operator and task (Figure 1.5). Operator-site input and robotic systems are known as the *master* while remote-site sensors and robotic systems are referred to as *slave* (Figure 1.5). The entire system is therefore known as a *master-slave system*. Bidirectional *master-slave systems* also provide force feedback in addition to measured motions. If the operator forgets about the interface because it is so effortless then the interface is referred to as transparent. A more thorough review of teleoperation can be found in [Siciliano and Khatib, 2008].

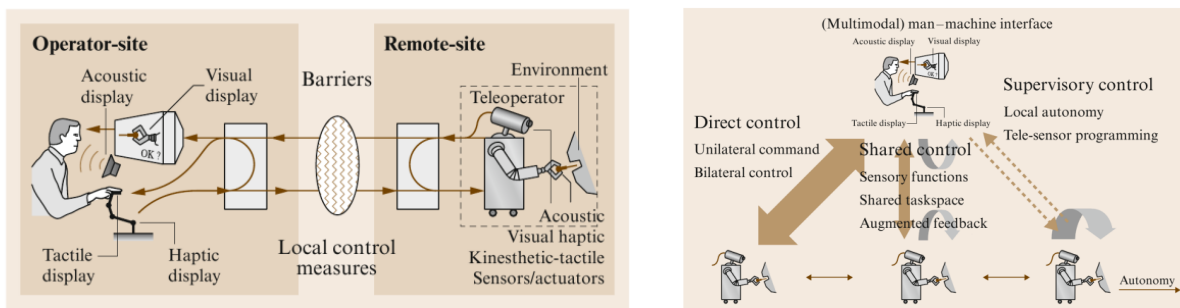


Figure 1.5: Telerobotic system overview (left) and different telerobotic control methodologies (right) [Siciliano and Khatib, 2008]

The necessity of handling radioactive materials during the 1940's led to the first major experiments in teleoperation through a barrier. The first work with bilateral simulators was at Argonne National Laboratory in the 1940's and 1950's by Goertz [Goertz, 1949]. During the late 1950's computers were integrated with mechanical systems. Over the past several decades telerobots have expanded from their roots in the nuclear industry and been developed for a wide variety of applications including hazardous environments, search & rescue, and space robotics. During this time a spectrum to describe the different control architectures was

developed (Figure 1.5). Throughout the 1960's interest in manual control for aircraft and cars increased. By 1967 the idea of higher level supervisory control had developed from research into human teleoperation of robots on the moon. Interest and development of supervisory and shared control grew in the late 1970's and 1980's and continues today [Chiou et al., 2015]:

- Direct (manual) control: operator is controlling the robot without any automated assistance
- Shared control: some degree of autonomy is available to the operator
- Supervisory control: only high-level operator commands and system feedback

The choice of coordinate frame for teleoperation is exceedingly important to the intuitiveness of the system. It has been suggested, with limited numerical support, Task-Centered (TC) coordinate frames are the most efficient. TC coordinate frames are like thinking of directly facing the task object and interacting with it from this viewpoint. In the study [Hiatt and Simmons, 2006] users only performed three operations but the mean completion time for the last operation was significantly faster for TC than View Point (VP) or Robot-Centered (RC). Thus, the performance gap between TC, VP, and RC frames will increase with increasing number of operations [Hiatt and Simmons, 2006].

1.2.4 Virtual Fixtures

Virtual Fixture (VF)s use is an important aspect of semi-autonomous behaviors and shared control [Rosenberg, 1993]. VFs are virtual constraints imposed on the user's workspace much like jigs used in machining or assembly modeling constraints. VFs can be constructed from a number of different geometric sources (Figure 1.6). Assistance levels can

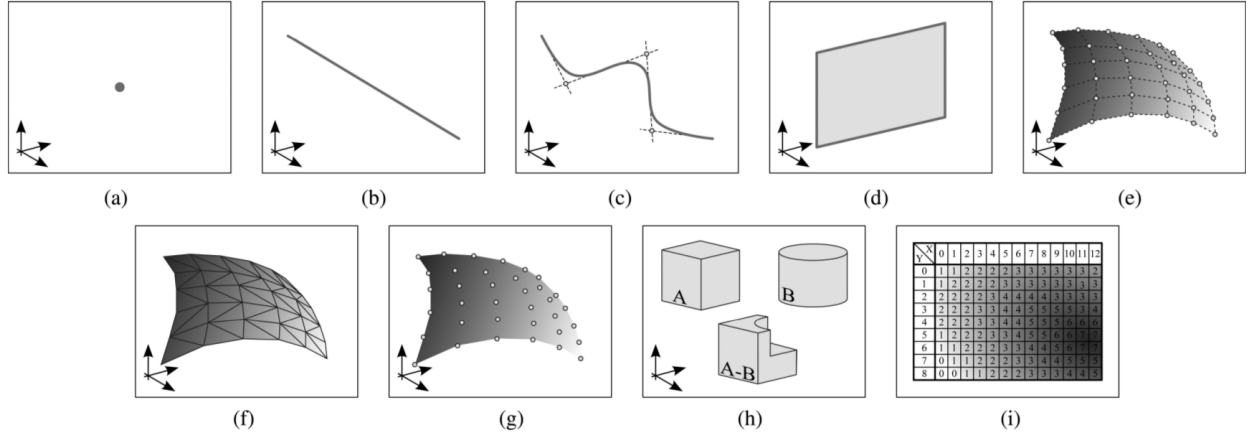


Figure 1.6: Example illustrations of the VF representations. (a) Point (b) Linear (c) Parametric curve (d) Planar (e) Parametric surface (f) Polygonal mesh (g) Point cloud (h) Volumetric primitive (i) Explicitly described. [Bowyer et al., 2014]

be preprogrammed and varied between tasks or users. Many VFs guide motion paths through corrective positions or forces. These type are called a Guidance Virtual Fixture (GVF). VFs can also assist task completion through exclusion zone workspace limitations which are known as Forbidden Region Virtual Fixture (FRVF). Operator control is therefore shared and task knowledge is utilized to capitalize on manipulator accuracy increasing safety and efficiency during teleoperation. Rosenberg states that “virtual fixtures can improve human-machine performance, while allowing the user to maintain ultimate control over the task execution.” [Abbott et al., 2007]

1.2.5 Robot Operating System

The chosen software for developing the proposed semi-autonomous behaviors for intuitive teleoperation is the Robot Operating System (ROS). ROS is an open-source software

used for motion planning, trajectory execution, navigation, and sensor integration. The software is modular in nature and emphasizes a high level of abstraction. It is presented as:

“a framework for writing robot software. It is a collection of tools, libraries, and conventions that aim to simplify the task of creating complex and robust robot behavior across a wide variety of robotic platforms.” [Quigley et al., 2015]

The greatest benefit of ROS is its collaborative and open source nature. It has become the *de facto* standard in robotics research with approximately 80 supported robots and over 2000 software packages [Quigley et al., 2015]. Rapid dissemination of cutting edge research source code through ROS packages allows debugging, expansion, and package redistribution by anyone in the community. In short, it allows a continual improvement and expansion process to exist without proprietary limitations. New applications can be written in either C++ or Python and communicate to existing nodes through the messaging structure.

“A system built using ROS consists of a number of processes, potentially on a number of different hosts. . . The fundamental concepts of the ROS implementation are nodes, messages, topics, and services. Nodes are processes that perform computation. . . A node sends a message by publishing it to a given topic. . .” [Quigley et al., 2009]

Once a description of the physical robot - a Unified Robot Description Format (URDF) file - is created, users are able to utilize a large number of published packages. This allows rapid expansion of high-level robot capabilities since low-level requirements are provided but can be tweaked if desired. There are packages for robot control, *MoveIt!* [Sucan and

Chitta, 2017a], visualization, *RViz* [Open Source Robotics Foundation, 2015], and various sensors including ‘ar_track_alvar’ [Scott Niekum, 2016] which allows a Red-Green-Blue-Depth (RGBD) cameras to localize barcodes in 3D space. The version of ROS used for this work is Indigo and it is used to control the TurtleBot, Pioneer, VaultBot, Yaskawa SIA5, and Yaskawa SIA10 hardware platforms.

The rapid development, open source nature of ROS, and use of industrial systems at the NRG are the main reasons the NRG joined the ROS-Industrial consortium [ROS-Industrial, 2015a; Edwards and Lewis, 2012]. ROS-Industrial (ROS-I) is an open-source project which seeks to expand the capabilities of ROS to industrial robotic systems for advanced manufacturing. They have led the development of ROS interfaces for many industrial controllers including Universal Robotics’ UR and Yaskawa’s SIA robotic manipulator lines [ROS-Industrial, 2015b]. The UR5 drivers used on the VaultBot were developed by ROS-I ensuring low-level hardware driver development need not be duplicated. A more thorough description of the SIAs is presented in Chapter 3. Additional ROS tools upon which the proposed work builds are briefly described below.

1.2.6 *MoveIt!*

The ROS package for manipulator motion planning is *MoveIt!*. It was developed to be easy for novice users to get started but allows detailed customization for experienced users and higher complexity hardware platforms:

“*MoveIt!* is state of the art software for mobile manipulation, incorporating the latest advances in motion planning, manipulation, 3D perception, kinematics, control and navigation.” [Sucan and Chitta, 2017a]

MoveIt! has been used on over 65 robots by the ROS community including manipulators built by Universal Robotics, Yaskawa, and KUKA [Sucan and Chitta, 2017b]. The previously mentioned robot description, URDF, is used to create a *MoveIt!* configuration package which includes information like link & joint names, joint limits, groups of joints which function together (planning-groups), a matrix of possible link collisions, and controller names.

By default, FK and IK calculations are performed by Kinematics and Dynamics Library (KDL) [The Orocos Project, 2015a] which was developed by the "Open ROBot Control Software" (Orocos) Project [The Orocos Project, 2015b]. The solver has been incorporated into *MoveIt!* as a plugin allowing custom solvers to be easily integrated. One such example is the 'ur_kinematics' package which was based on work by Kelsey Hawkins at Georgia Tech [Hawkins, 2013]. This package contains a custom IK solver for the 6 DOF UR5 and UR10 robots.

Motion planning capabilities are provided through the Open Motion Planning Library (OMPL) [Sucan et al., 2012]. OMPL has a variety of motion planners available including [The Open Motion Planning Library, 2015]:

- Probabilistic Road Map Method (PRM): LazyPRM, PRM*, LazyPRM*
- Rapidly-exploring Random Trees (RRT): RRT Connect, RRT*, Lazy RRT
- Expansive Space Trees (EST): SBL, pSBL

The Flexible Collision Library (FCL) is the primary collision checking package [Pan et al., 2012; GAMMA Group, UNC Chapel Hill, 2015]. One of the key concepts of collision checking is the Allowed Collision Matrix (ACM). Essentially the ACM is a matrix of binary

values relating whether or not collisions between links are even possible based on joint limits. If the matrix entry for two links is ‘1’, collision checking is unnecessary. This concept is implemented for the robot description in the *MoveIt!* configuration package as the ‘Default Self-Collision Matrix’.

The state of the robot and workspace are known as the ‘planning_scene’. This information can be stored in a MongoDB database management system (DBMS) and interactively displayed through ROS by using *RViz* (described next). *MoveIt!* can be used with simulated robots or on hardware, depending on the controller information specified in the configuration. Trajectory points are sent to and joint states are read from the hardware controller through the use of ROS messages with requirements set by the ‘moveit_controller_manager’.

1.2.7 *RViz*

RViz is a 3D robot visualizer for ROS which uses Open Source 3D Graphics Engine (Ogre3D) [OGRE, 2015]. It can display various types of data including robot configuration, sensor, and navigation data, among many others [Open Source Robotics Foundation, 2015]. The window can be customized by adding or removing panels from the display. Plugins can also be written when additional functionality is required. *RViz* allows integrated visualization of the robot, its environment, and published sensor data alongside robot control within a single window. RGBD camera data can be viewed as image or point cloud data, allowing checking of collision mesh locations. Collision meshes create exclusion volumes in the workspace for motion planning. Robots can be controlled through 6 DOF interactive markers [Gossow et al., 2011] in *RViz* by loading the *MoveIt!* configuration (Figure 3.6). This means master control is commanded through a virtual representation of the manipulator instead of more traditional

interfaces such as a joystick, slave manipulator, or other common manual controllers.

1.3 Objectives

The broad goal of this research is to decrease operator burden by enabling semi-autonomous behaviors. Therefore part - but not all - of the task is accomplished autonomously leaving the operator to manage - but not micromanage - a task's overall execution. When multiple levels of semi-autonomous control are available on a system, it is referred to as Levels of Autonomy (LOA). Variable autonomy applies when the operator can revert to lower (closer to manual) levels of control to improve their comfort with task execution. This research focuses on creating a high-level semi-autonomous behavior for a common subset of tasks relevant to non-contact but time-intensive tasks such as remote inspection (visual or radiological), laser cutting, and plasma cleaning. Alternative contact tasks include assembly, surface preparation (grinding, polishing), and swabbing surfaces for contamination testing. One burdensome aspect of many tasks is mentally accounting for the spatial transformations between the robot and its environment. Fortunately, environments dealing with radioactive materials as well as many inspection and manufacturing processes, are generally static in comparison to an *open world* environment. In many cases, sensor data and *a priori* knowledge of the task, tool, and environment are available. Tasks requiring a set EEF orientation and offset to the task surface which is known *a priori* provide additional constraints allowing task simplification. It is, therefore, possible to use the available information to precompute spatial transformations necessary for task completion and reduce operator mental burden. The primary focus of this research effort is to construct, previously uninvestigated, point cloud GVs based on task geometry and task execution parameters. These point cloud GVs

will extend VF construction methods by automatically generating more general, complex, flexible, and expressive VFs which will decrease the operator’s spatial transformation mental load during task execution. The generated VFs are necessarily more complex than current methods. Thus, a secondary objective is to evaluate their usability for a set of application tasks relevant to the motivating domain.

1.4 Organization of Dissertation

This chapter serves as an introduction to the complexities but necessity of adopting robotics in the nuclear domain. This chapter also included a primer of the core robotic research areas related VF generation and use.

Chapter 2 begins by reviewing robotic efforts utilizing semi-autonomous behaviors to complete nuclear tasks to establish their potential effectiveness in the domain. The chapter ends with a detailed analysis of the remaining research and technological gaps.

Chapter 3 discusses the combination of a task defined FRVF and GVF layers into a singular VF. These are preliminarily based on volumetric primitives and are applied to several spatially discrete and spatially continuous tasks. Selecting from precalculated poses reduced operator mental burden during task execution.

Chapter 4 extends combination VFs with a generation approach based on complex task geometry which forms the core of this effort (Figure 1.7). This geometric approach increases VF generality and flexibility to meet needs presented in the literature. TVF input data types and checks are discussed. VF generation algorithms are outlined and the graph structure output detailed.

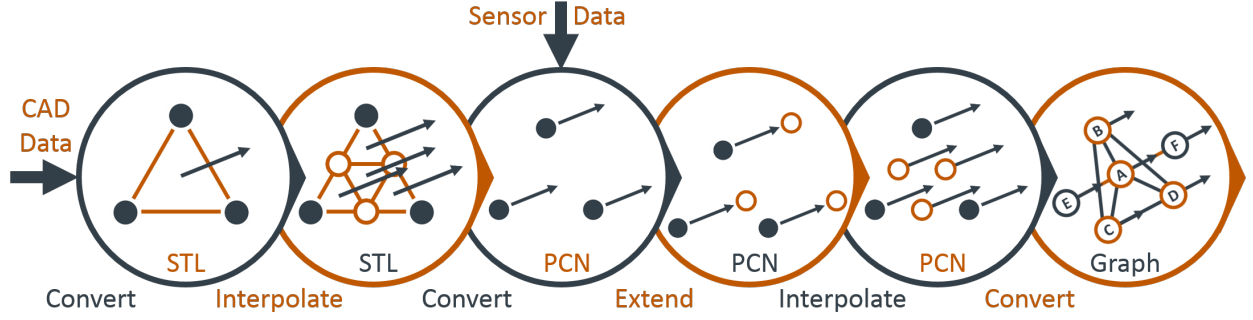


Figure 1.7: Task surface geometry based VF generation pipeline.

Chapter 5 implements the algorithms defined in Chapter 4 in C++ using a variety of third party software libraries. This chapter also examines visualization and task execution environments in *RViz* and RobotStudio along with their respective strengths. The developed software pipeline (Figure 1.7) will also be applicable to other domains such as emergency response.

Chapter 6 describes generated VF evaluations with multiple input data types. These tests show the strengths and limitations of the TVF generation pipeline. Spatially discrete and spatially continuous operator evaluation studies follow input variations. Then quantitative and qualitative results are discussed.

Chapter 7 contains a summary of the previous chapters and review of research contributions. Avenues of future VF expansion and use are also outlined.

Chapter 2

Literature Review

The previous chapter served as an introduction to teleoperation difficulties in nuclear robotics and similar application areas. This chapter reviews work related to the proposed semi-autonomous behaviors for addressing this problem. These areas of research include robots in nuclear facilities, semi-autonomous behaviors, virtual fixtures, and D & D tasks. The chapter concludes with an analysis of the remaining research and technological gaps.

There has been a slow but steady embrace of robotics technologies in the nuclear field. Beginning with static teleoperation systems and progressing to include mobile and mobile manipulation platforms. Examples include Sandia's M2 robot freeing a stuck radiation source [Laboratories, 2005], the development of small nuclear facility inspection robots [Guan et al., 2014], and procedures for making a robot for use in nuclear accidents [Ma et al., 2014].

2.1 Robots with Semi-autonomous Behaviors at Nuclear Facilities

The broad goal of this research is to decrease operator burden by advancing semi-autonomous behaviors. Therefore part - but not all - of the task is accomplished autonomously leaving the operator to choose the appropriate LOA to manage - but not micromanage - a task's completion.

A line of research with similar goals to the Nuclear Robotics Group's (NRG) at the

University of Texas at Austin is the Autonomous Robotic Inspection Experimental System (ARIES-DOE) platform [Byrd and Pettus, 1996], [Byrd, 1996]. ARIES-DOE was a mobile inspection robot for the DOE and should not be confused with the Advanced Recover and Integrated Extraction System (ARIES-LANL) at LANL [Wood, 1996]. It was designed to inspect steel drums containing mixed and low-level radioactive waste for exterior damage and corrosion. These drums (55, 85, and 110 gallon varieties) are stacked four high in DOE warehouses (Figure 2.1). The goal of ARIES-DOE was to:

“relieve the warehouse inspector of the tedious, mundane, and potentially radioactive exposing task of inspecting barrels. . . Cost savings, **reduced worker radiation exposure**, improved documentation, improved quality with inspection consistency. . . are some of the anticipated benefits from autonomous inspection” [Byrd, 1995]

Unfortunately, publications related to the ARIES-DOE program stop in 1997 suggesting the program was discontinued. Their application was described as:

“more challenging than originally anticipated. The fact that individual subcomponents have been proven in the field does not imply that a machine full of them will work reliably” [Hazen, 1996]

Several other technological limitations such as “laser based inspection and drum centering do not work in bright light conditions” [Hazen, 1996] were also mentioned. Technological advances from the last several decades should make the application of mobile robotic technologies for inspection tasks a feasible path forward.



Figure 2.1: ARIES-DOE stowed (left) and navigating through a three foot aisle [Byrd, 1996].

The use of mobile robots with LOAs at national laboratories can be traced back through a semi-autonomous surveying system developed by Bruemmer et al. [Bruemmer et al., 2002] at Idaho National Laboratory (INL) He noted a:

“great potential benefit in introducing robotic automation to nuclear facilities. The replacement of human labor with machine systems promises to improve personnel safety by **reducing total man-hours in hazardous areas**” [Bruemmer et al., 2002]

The research was performed on an iRobot mobile platform with a sensor suite and

custom interface software (Figure 2.2) but no attached manipulator. During this research a framework for levels of shared autonomy to increase user comfort for mobile systems was developed (Table 2.1). In the framework, more autonomy means the number of behaviors (preventing collision, etc.) handled autonomously also increases. This process continues until tasks are no longer completed semi-autonomously but with full autonomy. The authors present an impressive efficiency increase and dosage reduction (3 days/7 mRem compared to 3 months/82mRem). They also conclude several areas need additional investigation. These topics were not technical in nature, but rather focus on the operator robot interactions, including operator trust, job satisfaction, and training re-alignment.

Table 2.1: Layered Autonomy Modes for Mobile Platforms

Mode	Defines Tasks	Motivates Motion	Prevents Collision
Teleoperation	Operator	Operator	Operator
Safe	Operator	Operator	Robot
Shared	Operator	Robot	Robot
Autonomous	Robot	Robot	Robot

As an investigation into some of these topics the NRG developed similar variable autonomy modes for remote radiation surveying with an inexpensive mobile system [Sharp and Pryor, 2015]:

- **Teleoperation:** allows direct, continuous user commands.
- **Safe:** adds collision detection during teleoperation. Collision objects can be inferred from models of the environment, real-time sensor data, or contact dynamics as shown in [Schroeder et al., 2013].
- **Shared:** sends the robot to where the user desires via a point & click interface with autonomous way-point navigation.

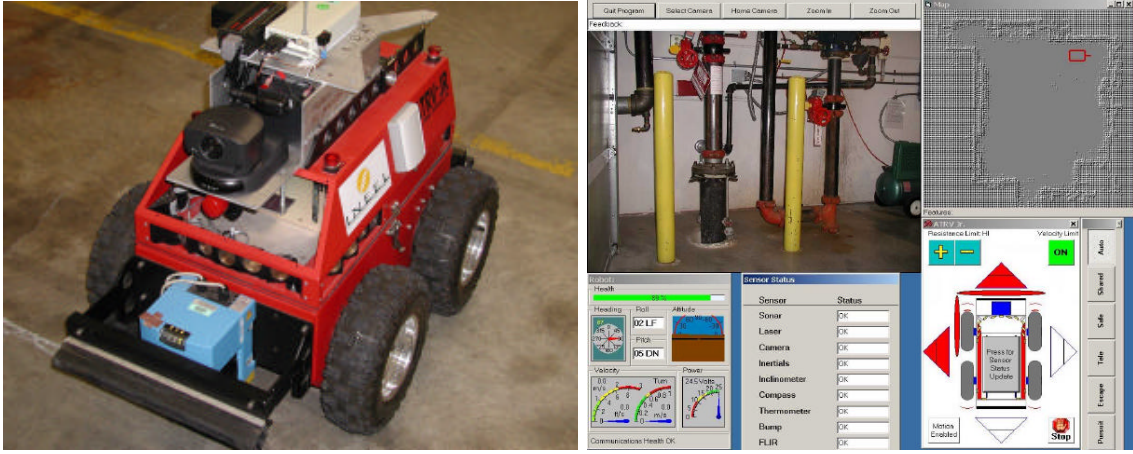


Figure 2.2: INL's early iRobot Robotic Platform used for radiation inspection and mapping (left) customized interface used in early efforts at INL to autonomously inspect contaminated environments (right) [Bruemmer et al., 2002].

- **Collaborative:** allows the user to command the robot to autonomously navigate to known target locations (i.e. "Go to table one.").
- **Autonomous:** user input is only required to start survey algorithm.

The results of the NRG's investigation show a slight increase in the system's ease-of-use rating for higher levels of autonomy. Overall however, the small testing pool, simple task, and safety of operating in a simulated environment limits the significance of these results [Sharp and Pryor, 2015]. Moving forward, more rigorous testing methods will be required.

The Advanced Recover and Integrated Extraction System at LANL (ARIES-LANL) also shows advancement of robotics technologies in the nuclear domain [Turner et al., 2009]. The more recent of several generations of upgrades to this system have been designed to function with semi-autonomous behaviors. Specifically, repetitive actions in the Disassembly Module such as retrieving and exchanging grippers are preprogrammed. Also, the Robotic

Integrated Packaging System (RIPS) Module which handles material canning and decontamination is capable of autonomous, semi-autonomous, or manual fault recovery. Thus the RIPS Module demonstrates multiple LOAs. The authors conclude there is:

“...considerable opportunity for improved nuclear material handling and processing capabilities with reduced occupational radiation exposure is made possible by automation technologies. Further development of key technologies will lead to future generations of automation systems with **enhanced intelligence**, improved state-awareness...” [Turner et al., 2009]

An additional LANL robotic system presented in [Turner et al., 2009; Harden and Pittman, 2008] was designed to clean out spherical containment vessels and also possesses several LOAs. This system is capable of manual, semi-autonomous, and automatic modes of operation. The controllers also have a number of different operating frames including global, spherical, and EEf frame Cartesian.

Another, more recent, contribution to mobile robotics for nuclear applications is the use of robots at the Fukushima Daiichi Nuclear Power Plant. On June 24, 2011, the first mission of a customized Quince mobile platform was conducted in the reactor building of Unit 2 [Nagatani et al., 2013]. Six missions were completed by the platform but it was unable to return from the last and was abandoned. From their experiences, system developers documented the critical lessons learned from the project:

- A lack of precise communication between researchers, developers, and operators
- A lack of education of operators in simulated environments
- A lack of field knowledge for researchers

Recently, operators were trained on NRG’s VaultBot mobile manipulation platform to complete a pipe inspection task at the DOE Office of Environmental Management’s (EM) “EM Science of Safety: Robotics Challenge” Aug. 22-25th 2016 at the Portsmouth Gaseous Diffusion Plant. The goal of operator interaction and demonstration highlighted EM’s belief in robotic enhancement of worker health, safety, and performance as robotics research organizations from all over the country were invited. Rodrigo Rimando, director of EM’s Office of Technology Development, noted:

“Structuring the demos to have the workers and operators conduct the demos provided us a unique opportunity to gain their perspectives on the **utility of the technologies** and to offer their insights on ways to make their work safer and easier to do” [DOE, 2016]

This event helped to address two of the key lessons learned from [Nagatani et al., 2013] by developing lines of communication between researchers and operators to encourage the utility of future robotic research to nuclear domains.

A recently published work which also builds upon [Bruemmer et al., 2002] is [Chiou et al., 2015]. Their experiment involved LOAs for navigation of a maze-like environment with a secondary user task. Overall, the number of system collisions were highest with teleoperation but completion times were significantly higher with autonomous navigation. The authors do note the “secondary task proved inappropriate, mainly because of its unclear impact on performance”. Another significant contribution of the publication is a set of guidelines for future experiments of this type:

- Tasks must require teleoperation and autonomy to be better or successfully completed.

- Extensive participant training makes experiments time consuming but ensures trust in the autonomous system.
- Situational awareness affects performance but is very complex to measure.
- Operator workload is difficult to measure in real time without using physiological techniques.
- Degraded performance, jointly with context, might be simplified by using experiments with ‘idle time’ as a performance metric.

2.2 Additional Robots with Levels of Autonomy and Evaluation Criteria

Other guidelines of importance can be drawn from [Yanco et al., 2015] where they analyze eight of the 15 DARPA Robotics Challenge Trial teams. The authors analyzed the competition based on team success, critical incidents, team utterances, subtasks, interface displays, operators / input devices / screens, and control methods. The results reinforced the currently held guidelines of “more sensor fusion, fewer operators, and more automation lead to better performance” [Yanco et al., 2015]. More specific suggestions included effectively fused data streams reduce cognitive loads over data streams in separate windows. Also, having too many operators can make it difficult to manage information and maintain situational awareness. It is important to note the highest scoring team displayed everything on one monitor. These guidelines are utilized in the operator evaluation of this research’s final system.

LOA approaches have been used with increasing complexity and improve task completion time and accuracy. The RoboSimian, used in the DARPA Robotics Challenge, for

example had basic behaviors such as walking to a designated location, grasping an object, and performing manipulation [Hebert et al., 2015]. In [Hentout et al., 2013] semi-autonomous behaviors and an integrated GUI led resulted in a 425% faster task completion.

In [Ciocarlie et al., 2012] semi-autonomous behaviors were combined with a non-roboticist operator for completion of various tasks in a cluttered home environment. This work illustrates many of the advantages of semi-autonomous behaviors including performance gains. Among the lessons learned, the greatest impediment to widespread application was the difficulty in accounting for the complexity of home environments. Luckily the structured nature of LANL nuclear facilities and other DOE environments significantly reduces these concerns.

2.3 Virtual Fixtures

An important aspect of semi-autonomous behaviors, shared control, and this research is the use of Virtual Fixtures (VF) [Rosenberg, 1993]. VFs are virtual constraints imposed on the user’s workspace much like jigs used in machining or assembly modeling constraints. Assistance levels can be preprogrammed and varied between tasks or users. Many VFs guide motion paths through corrective positions or forces. These type are called a Guidance Virtual Fixture (GVF). VFs can also assist task completion through exclusion zone workspace limitations which are known as Forbidden Region Virtual Fixture (FRVF). Operator control is therefore shared and task knowledge is utilized to capitalize on manipulator accuracy increasing safety and efficiency during teleoperation. Rosenberg states that “virtual fixtures can improve human-machine performance, while allowing the user to maintain ultimate control over the task execution.” [Abbott et al., 2007]

A recent and comprehensive review of VFs and active constraints can be found in [Bowyer et al., 2014]. Over 120 publications are reviewed where the vast majority pertain to teleoperated or hands-on master-slave systems with haptic feedback. The authors outline a generalized VF framework consisting of a VF generator, VF controller, and robot hardware. Several VF generation options are presented:

- Point - (Figure 2.3.a)
- Linear - (Figure 2.3.b)
- Parametric curve - (Figure 2.3.c)
- Planar - (Figure 2.3.d)
- Parametric surface - (Figure 2.3.e), generalized testing sources
- Polygonal mesh - (Figure 2.3.f), online databases for crowdsourcing
- Point cloud - (Figure 2.3.g), RGBD sensing for “open world” scenarios
- Volumetric primitive - (Figure 2.3.h), initial methodology (Chapter 3)
- Explicitly described - (Figure 2.3.i)
- Artificial neural network constraints

There are a variety of VF generation techniques presented, but recent efforts have focused on haptic surgical domains, [Castillo-Cruces and Wahrburg, 2011; Rydén et al., 2011; Kosari et al., 2014]. In [Rydén and Chizeck, 2012], RGBD data is preprocessed and used for direct reconstruction of the point cloud surface. Each protected point is then assigned a FRVF radius and stiffness proportional to region penetration. Motion is an iterative determination of system states to allow movements either on or away from an estimated plane. Haptic forces are then fed back to the master control and the new slave position is set. The system

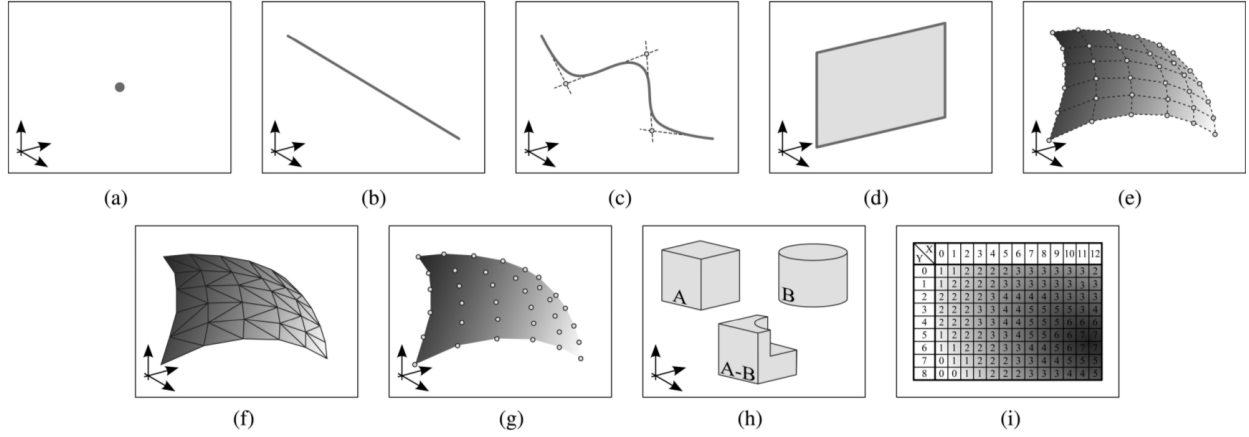


Figure 2.3: Example illustrations of the VF representations. (a) Point (b) Linear (c) Parametric curve (d) Planar (e) Parametric surface (f) Polygonal mesh (g) Point cloud (h) Volumetric primitive (i) Explicitly described. [Bowyer et al., 2014]

is operated with a Kinect sensor at 30 Hz. Another publication, [Yamamoto et al., 2012], also uses RGBD data to generate surfaces but the FRVF is offset by a specified amount from the detected surface. Reaction forces are computed and displayed for the user through a material property overlay. This allows the user to search for changes in tissue stiffness which might be caused by cancerous tissue. These two publications and one more, which is based heavily on Rydén et al., [Nia Kosari et al., 2014], all convert point cloud information directly into a FRVF. In several cases the variability of the human body led researchers to create polygonal mesh FRVFs from sensor data [Ren et al., 2008; Li et al., 2007]. Some research used point cloud FRVFs [Kosari et al., 2014] but none of the surveyed papers use point cloud information for GVF generation.

The correlation between surgical haptics and VF use seems to be linked to the need to protect critical areas from damage. Ren et al. state “Virtual fixtures can also help to

constrain the surgical tool within the confined area, thus preventing accidental contact with sensitive tissue near the surgical site” [Ren et al., 2008]. In many nuclear material operations, automated system failure requires personnel to complete the task manually [Turner et al., 2009] or repair the equipment [DeJong et al., 2006]. As with patients in the surgical domain, personnel entry into hazardous environments presents a risk and both cases can be referred to as *high-value* operations.

The review also discusses several constraint evaluation metrics such as simple constraint representations, spatial partitioning, bounding volume hierarchies, and feature-tracking algorithms. The proposed research will also use simple constraint representations specifically collision avoidance while planning and software defined pose tolerance values. Also reviewed are constraint enforcement methods including:

- Simple Functions of Constraint Proximity
- Proxy and Linkage Simulation
- Nonenergy Storing Constraints
- Potential Fields
- Reference Direction Fixtures
- Constrained Joint Optimization
- Passive Constraint Enforcing Mechanisms

Constraint enforcement may not directly carry over as well as some of the other metrics since the industrial systems often lack haptic feedback. In conclusion, Bowyer et al. make several observations on current challenges [Bowyer et al., 2014]:

- “flexible structures such as **meshes and point clouds** allows the constraint generator... to be much **more expressive**, making constraints of **greater use...**”
- “The most significant challenge now seems to be the **initial step of generating the constraint geometries** in an effective way...”
- “future research into active constraints will move toward a **more complete representation** of the working environment...”

VFs have also been investigated for use in nuclear applications. At Argonne National Laboratory (ANL) VFs were applied to a D & D task of cutting a pipe into sections [DeJong et al., 2006]. The VF limited the motion of the EEF, a rotary saw in this case, to a plane perpendicular to the pipe axis. Motion is allowed up to and around but not along or into the pipe (Figure 2.4). This approach effectively created a FRVF taking up all space with the exception of the plane perpendicularly approaching the pipe. This limitation on EEF motion reduced the stresses on the cutting blade and thus reduced chances for damage or failure. In order to replace damaged or broken blades, personnel had to don PPE and enter the restricted area causing significant amounts of downtime. The PPE also then becomes Low Level Waste and requires expensive disposal. During testing the outcome showed manual operation resulted in 9.9° rotation out of the plane while operation assisted by the VF resulted in only 0.1° rotation. The VF also kept the operator from overshooting the surface and banging the saw into the pipe. Another feature allowed operators to place additional VFs in order to create additional FRVFs for case by case workspace restrictions [DeJong et al., 2006]. More recently Park et. al have continued this work by utilizing advances in augmented reality

and 3D sensing and modeling technologies [Park et al., 2014]. However, this is preliminary due to a lack of system testing results.

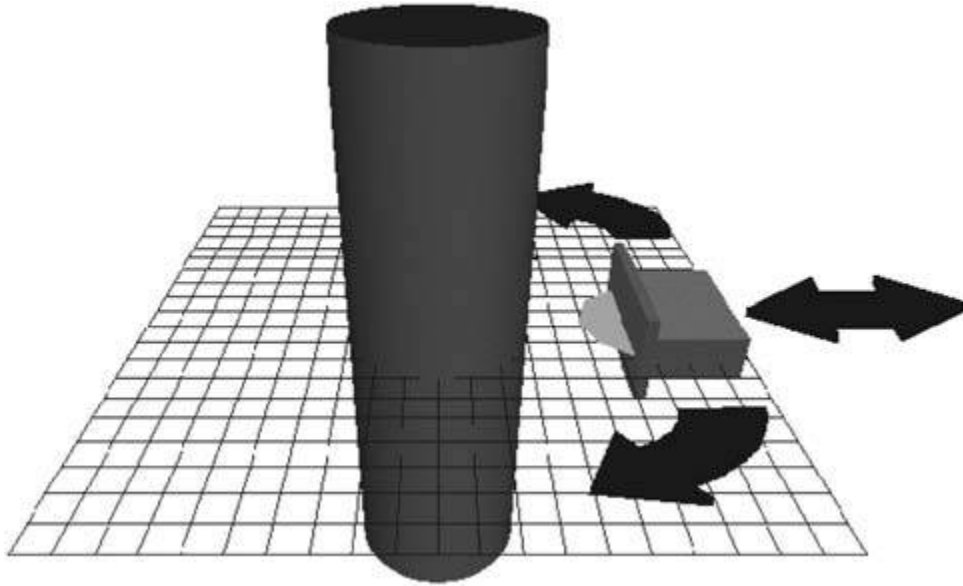


Figure 2.4: Argonne’s VF. The saw is constrained to be on the plane outside the pipe with arrows in the figure representing allowed motions [DeJong et al., 2006].

Another recent example of VF use and comparative operator evaluation took place at the NRG [Kruusamae and Pryor, 2016]. In this work simple workspace limitations (linear and planar) were paired with a natural language and hand gesture interface for EEF control in *RViz*. The VFs could be turned on and off with voice commands. A testing methodology of a short training period followed by multiple needle threading trials was implemented by Kruusamäe to collect task completion times. The resulting completion times demonstrated the new interface was highly intuitive, capable of high precision tasks, and achieved similar average task completion times to ROS interactive markers (86 sec (SD=42 sec) vs. 94 sec (SD=61 sec) respectively [Kruusamae and Pryor, 2016]). This experience at the NRG with

comparative evaluations of operator EEF control methods was valuable when setting up comparative evaluations procedures for this research effort.

Recently the United Kingdom’s Sellafield facility, which is the largest and most hazardous site in the Nuclear Decommissioning Authority’s estate, has had total lifetime costs for decommissioning the site reach £67.5 billion and is rising [House of Commons Committee of Public Accounts, 2013]. As such, the Sellafield Decommissioning Program has investigated the LaserSnake system for size reduction of large tubes and canisters (Figure 2.5). Khan and Hilton discuss the benefits of using fiber optic laser systems where the expensive laser generators can be kept away from contaminated materials [Khan and Hilton, 2013; Hilton and Khan, 2014]. This process is considered a thermal size reduction/dismantling technique as compared to a mechanical (sawing, shearing, milling) or hydraulic (water jet or abrasive water jet) technique. The key purpose in all of these techniques is to reduce the volume of contaminated materials and thus reduce the cost of disposal. Another benefit is a narrower kerf (i.e. the slit width made by a cutting operation) than other thermal cutting techniques.

Several tasks of varying complexity were presented in [Khan and Hilton, 2013; Hilton and Khan, 2014]. Demonstrations included including planar laser scabbling of a concrete section, linear “gouge” cutting with a “side” gas jet, and highly complex pipe manifold dismantling with linear cuts. Therefore, as radioactive waste often comes in complex shapes, most robotic systems lack the flexibility to intelligently adjust to unique pieces without reprogramming. In the work of Khan and Hilton, pipe cutting was limited to a single or double pass with a linear motion [Khan and Hilton, 2013]. According to Hilton and Khan:

“Future decommissioning of nuclear facilities will make increasing use of **non-contact** remote cutting techniques... ...What is needed is a highly automated



Figure 2.5: LaserSnake demonstration [Khan and Hilton, 2013].

remote technology that can deliver a non contact smarter dismantling process, cut most materials, cut **complicated structural geometries...**" [Hilton and Khan, 2014]

Teleoperation is an alternative to linear cuts but can be costly and time consuming to train operators. Furthermore, the task may still fail as noted by DeJong et al. [DeJong et al., 2006]. For such tasks, semi-autonomous behaviors can assist users with task completion while they maintain control of the overall task execution. However, current methods lack the capability to adjust to **complicated structural geometries**.

Another body of work which relates to this research is machine tool path generation.

While working towards a more defined goal, the fundamental calculations are very similar. For example in [Xiuzhi Qu and Brent Stucker, 2003] the authors present a method of offsetting 3D surfaces for STL models. Additional papers such as [Hatna et al., 2000; Hsi-Yung Feng and Huiwen Li, 2002] perform similar calculations for shrinking or expanding CAD models but focus on tool scalloping and other machining concerns.

2.4 Literature Review Summary

The broad goal of this work is to decrease operator burden by advancing semi-autonomous behaviors. Previous mobile robotic applications in nuclear domains met with limited success [Hazen, 1996; Nagatani et al., 2013] but technological advancements and the development of stable support software over recent years provide a footing for further advancement. The benefits of LOAs and VFs were reviewed and have “great potential benefit” [Bruemmer et al., 2002] and can “improve human-machine performance” [Abbott et al., 2007]. VF benefits have been extended to a nuclear environment by [DeJong et al., 2006] but other research suggested operator education [Nagatani et al., 2013] and comfort level [Bruemmer et al., 2002] concerns with LOAs. Increased operator training in simulated [Sharp and Pryor, 2015] and demonstration [DOE, 2016] environments show a reduction of such concerns.

VFs have been mainly used in haptic surgical domains, and additional research is necessary to transfer techniques to additional domains. Haptic surgical domains seem to have been the focus of VF research due to the need to protect critical areas during surgery which can be referred to as *high-value* operations. Similarly, LANL tasks involving *high-value* operations may be ideal test beds for VF research advancement and application to industrial hardware.

The focus of this research is to address VF generation shortcomings noted in the literature “[t]he most significant challenge ... the initial step of **generating the constraint geometries** in an effective way...” [Bowyer et al., 2014]. This process will provide a “more **complete representation** of the working environment” [Bowyer et al., 2014] and fulfill the need for “a highly automated remote technology that can deliver a non contact smarter dismantling process, cut most materials, cut **complicated structural geometries**...” [Hilton and Khan, 2014]. Broad applicability for the tasks and domains identified requires VF generation from parametric surfaces, polygonal meshes, CAD models, and point cloud sensor data. Thus, to complete this goal, meshes and point clouds are of the most interest for VF representation since, “flexible structures such as **meshes and point clouds** allows the constraint generator... to be much **more expressive**” [Bowyer et al., 2014]. Previous research used a point cloud for a single layer FRVFs instead of layers of point cloud GVFs. The product of this research, robust and hardware agnostic VF generation algorithms, will provide robots with more complex synthetic world models and enable improved semi-autonomous behaviors to assist with task execution.

Chapter 3

Preliminary Task Virtual Fixture Development

The goal of this research is to advance semi-autonomous behaviors in order to decrease operator burden since robotic systems will, ideally, be operated by trained technicians without expert intervention¹. Two significant barriers to adopting robotics and remote systems are, first, the *correspondence problem* where differences between operator and manipulator kinematics complicate motion plan anticipation; and second, translating to/from the input, output, and possibly intermediary contexts of the operator/robot interface, which is referred to as *context switching* and incurs a significant operator mental load. An example is when the operator must account for coordinate transformations between the operator & world frame, the world & manipulator frames, and the manipulator & tool frames while also maintaining the sensor offset from the surface. Task surface complexity exacerbates the issue. The simplest form of this process is placing a tool in the proper location relative to a flat task surface aligned with the global frame, such as laser scabbling [Khan and Hilton, 2013], which is not particularly difficult. A more difficult example is a cylindrical task surface such as a plane fuselage (Figure 3.1, left) or task surface with a constant cross-section with which to align (Figure 3.1, right). Operator mental load increases further with spherical tasks (Figure 3.2). As surface complexity increases further to contain multiple convex or concave regions, error

¹This chapter includes material published in Sharp and Pryor [2016]; Sharp et al. [2017b, 2018] where I contributed volumetric primitive TVF generation, demonstration set up, and results analysis.

chances during task completion increase significantly or task completion becomes untenable. This chapter outlines the general construction and combination of multiple types of VFs into a singular VF which more thoroughly describes the task than current methods. Environmental information, manipulator description, and tool geometry are necessary to construct a virtual manipulator workspace including collision obstacles for actual task execution. However, this information is hardware-centric and not considered during VF advancement efforts.

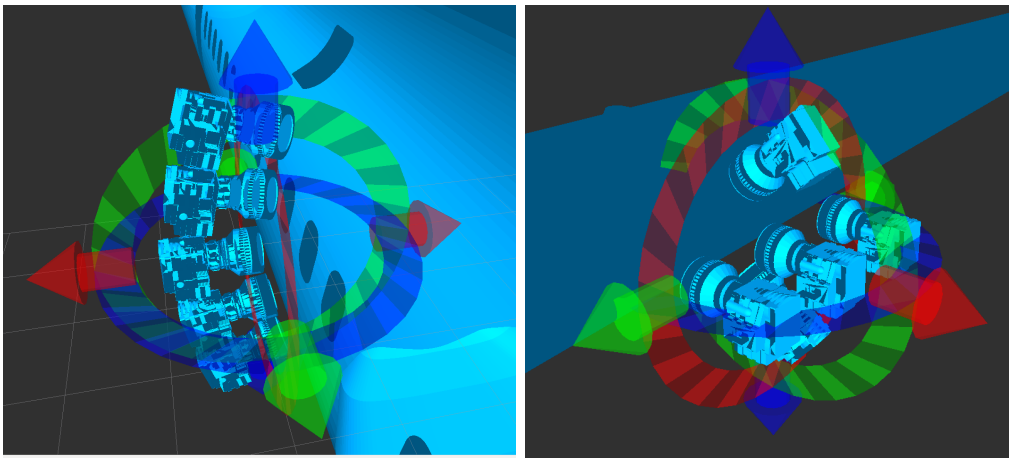


Figure 3.1: Cylindrical inspection of an airplane fuselage (left) and inspection locations for the complex surface of an airplane wing (right) in *RViz*.

This effort’s primary focus is to produce generalizable, hardware agnostic, VF generation algorithms which provide a more complete representation for non-contact task execution. While radioactive waste often comes in complex shapes, the environments are generally static in comparison to an *open world*. In addition, many tasks, such as inspection and laser dismantling, require an EEF orientation and offset to the task surface which is defined *a priori* [Khan and Hilton, 2013; Hilton and Khan, 2014]. Examples include maintaining focal distance for cameras and laser operations or path width in plasma cleaning and painting

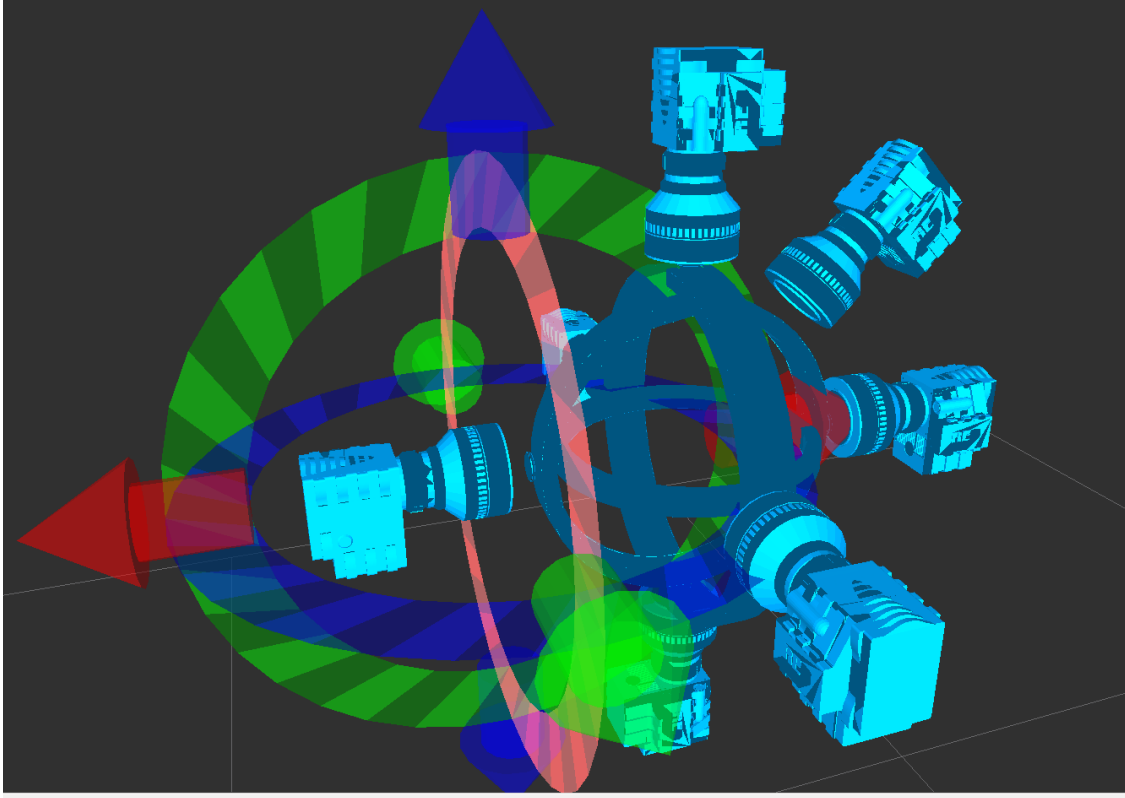


Figure 3.2: An example of camera positions around a spherical object in *RViz*.

(Figures 3.1, 3.2). As such, task execution requires operators to alter the transform to the surface along the surface (Left/Right, Up/Down) more than increasing or decreasing the distance to the surface (In/Out). Thus, task execution would be completed most effectively if the EEF were constrained to a surface offset from the task surface. The normal vector $\hat{\mathbf{n}}$ at a task surface location (Equation 3.1) can be calculated, selected as the ‘depth’ (In/Out) direction, and used to maintain surface distance during task execution. Therefore, we can use this knowledge of the task surface, requirements, and constraints to aid in VF generation, expand VF applicability, and use VFs to reduce the operator’s mental load.

$$\begin{aligned}
P_{surf}(i).v &= \begin{bmatrix} v_x & v_y & v_z \end{bmatrix} \\
P_{surf}(i).\hat{\mathbf{n}} &= \begin{bmatrix} n_x & n_y & n_z \end{bmatrix}
\end{aligned} \tag{3.1}$$

Multiple layers of offset surfaces can be created for task surface approach and task completion to maximize the VF's assistance to operators. Offset layers are generated along a task surface location's $\hat{\mathbf{n}}$ starting at a task-parameter-defined distance, d_{min} , and be placed at intervals, d_{inter} , until a maximum distance, d_{max} is reached. These parameters are independent of task surface geometry. Therefore, the same surface can have multiple VFs set up for different tasks or parameters for the same task can be applied to multiple surfaces.

Each offset layer is an individual VF and can be used as either a Forbidden Region Virtual Fixture (FRVF) or a Guidance Virtual Fixture (GVF). Either the task surface or a VF layer can be used as the task FRVF to minimize collisions during task execution. The FRVF and GVF layers are then combined into a singular representation called a Task Virtual Fixture (TVF). TVFs provide a more complete representation and heavily constrict the volume for task execution. Thus, TVFs allow the operator to maintain control, switching between GVF layers to utilize the best for the current portion of the task. TVF use is similar to the implementation at Argonne [DeJong et al., 2006] where the EEF is restricted to a surface. However, instead of restricting motion perpendicular to the surface to ensure controlled interaction, a TVF's surfaces are parallel to the task surface to limit interaction. This approach greatly reduces the possibility of undesired actions which could result in personnel having to don PPE to enter the environment and restore operations.

The combination of a task-defined polygonal mesh FRVF and point cloud GVF layers into a TVF forms the basis of this effort's developments. Multi-layered point cloud

GVFs were lacking in the previous approaches surveyed as is pairing polygonal meshes and point clouds into a single VF. The next section discusses preliminary volumetric primitive TVF development, implementation, hardware testing, and limitations. Volumetric primitive shortcomings lead into Chapter 4 where TVFs are generated from task surface geometry, which forms the bulk of this effort’s contribution.

3.1 Volumetric Primitive Task Virtual Fixtures

In many cases, a task surface and parameters allow the task surface to be treated as a volumetric primitive. In these cases, the task parameters also must allow VF layers to be constructed sufficiently far from the task surface for small surface features to be irrelevant. Task surface examples include walls, storage cans, and straight sections of pipe. Therefore, preliminary TVF investigations developed planar and cylindrical volumetric primitive TVFs. These volumetric primitive TVFs are referred to as Variable Normal Surface Virtual Fixtures (VNSVF).

Task parameters, minimum offset distance (d_{min}), distance between layers (d_{inter}), and maximum offset distance (d_{max}), are required to generate GVF layers at appropriate distances (Figure 3.3 top labels). Additional task parameters are required to construct VNSVFs. The task primitive type (planar or cylindrical) and task size (height = TS_h , width = TS_w , depth = TS_d or height = TS_h , radius = TS_r) are necessary to generate the FRVF task surface model (Figure 3.3 gray surface).

Since surface normals are known for volumetric primitives, VF layers are essentially larger versions of the task surface shape primitive (Figure 3.3). Thus, the first step in VNSVF construction is to calculate the number of VF layers using d_{inter} (Equation 3.2).

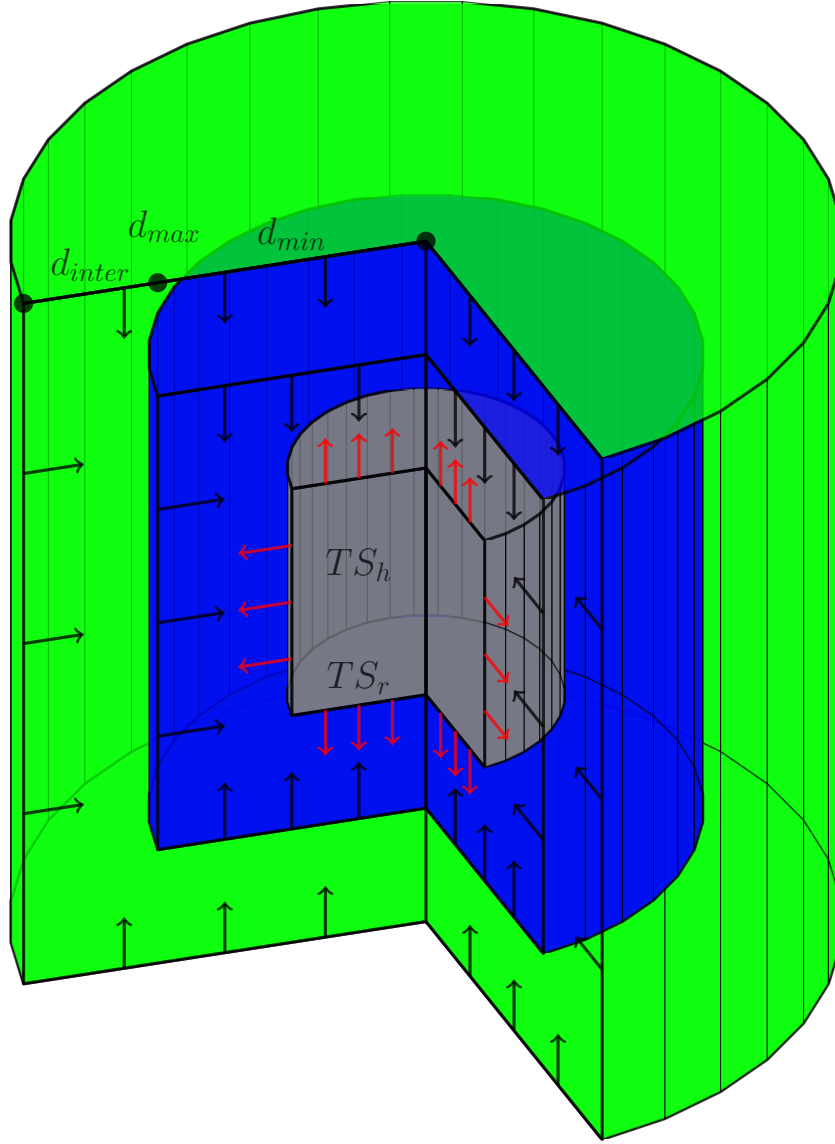


Figure 3.3: Visualization of a cylindrical (height = TS_h , radius = TS_r) volumetric primitive task surface (gray), its normals (red arrows), an offset VF surface at $d_{min} = 2 * TS_r$ (blue), and an offset surface at $d_{max} = TS_r$ (green) which is separated from d_{min} by $d_{inter} = TS_r$. Offset VF surface normals (black arrows) face back towards the task surface.

Distances between poses in the VFs are specified in the task parameters, d_{intra} . Therefore, the number of points in each direction of the array is calculated for planar (Equation 3.3) or cylindrical (Equation 3.4) volumetric primitives. Increasing distance from a cylindrical volumetric primitive will increase the number of poses in the layer to maintain layer resolution (Equation 3.4). After the number of points in the VF layer array is determined, VF poses are calculated in the surface frame using simple geometric relationships (Algorithm 1).

$$layer_{values} = int(\frac{d_{max} - d_{min}}{d_{inter}}) \quad (3.2)$$

$$x, y, z_{values} = int(\frac{TS_{h,w,d}}{d_{intra}}) \quad (3.3)$$

$$\theta_{values} = int(\frac{2 * \pi * (TS_r + layer * d_{inter})}{d_{intra}}) \quad (3.4)$$

Algorithm 1 Cylindrical GVF Algorithm

```
1: for layer in layer_values do
2:    $r = TS_r + layer * d_{inter}$ 
3:   Calculate the number of  $\theta_{values}$  (Equation 3.4)
4:   for z in z_values do
5:     for  $\theta$  in  $\theta_{values}$  do
6:       Calculate position:
7:        $pose.position.x = r * \cos(\theta)$ 
8:        $pose.position.y = r * \sin(\theta)$ 
9:        $pose.position.z = z$ 
10:      Calculate orientation facing back towards surface:
11:       $pose.orientation = \text{convert euler angles } (0.0, 0.0, \theta - \pi) \text{ to quaternion}$ 
12:      Save pose:
13:       $pose\_array.append(pose)$ 
14:    end for
15:  end for
16: end for
17: return pose_array
```

3.2 Volumetric Primitive Implementation and Evaluation

During implementation VNSVFs were constructed offset from the task surface region of interest (Figure 3.4) instead of surrounding the entire surface (Figure 3.3). This was done to limit Virtual Fixture interference and operator actions with large objects extending beyond the area of interest, such as wall sections and long pipes.

VNSVFs were implemented in ROS with *MoveIt!* providing robot control and *RViz* providing a visualization and control interface (Figure 3.5). One key difference to real-time teleoperation systems typically evaluated by DOE is EEF jogging is not well supported by ROS and ROS-I. Therefore, GVF layers were converted to arrays of ROS poses in the task frame (Figure 3.5 left, red markers) for position control. Therefore, only the world to VF

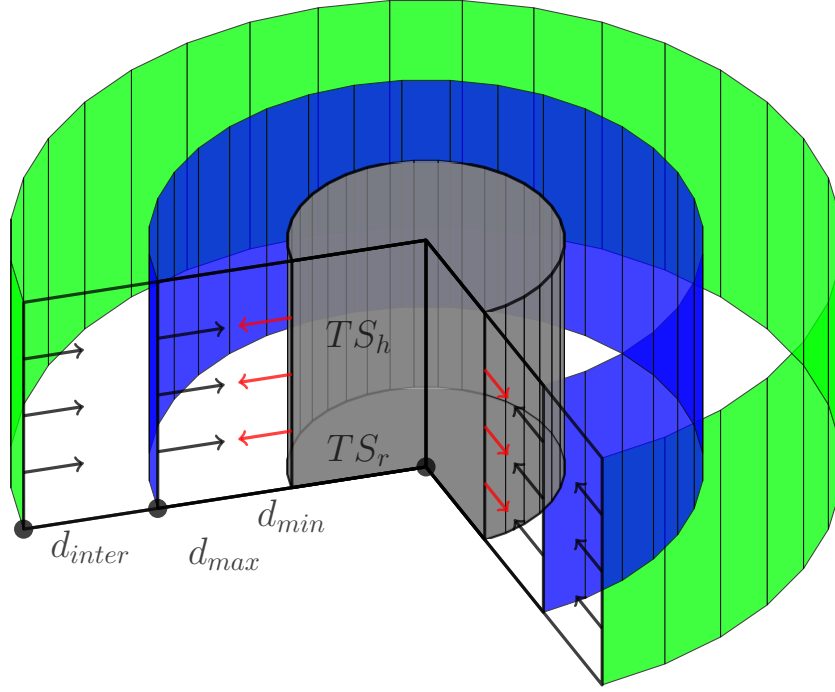


Figure 3.4: Visualization of a volumetric primitive VF implementation. VF layers (blue and green) and their surface normals (black arrows) are only calculated for the task surface (gray cylinder, height = TS_h and radius = TS_r) region of interest.

frame transform is required to place the VNSVF in the virtual workspace. This transform is defined when the task surface is located in the real world. A task-defined FRVF is loaded into the *MoveIt!* planning scene as a collision mesh (Figure 3.5 left, green cylinder).

In addition to the VNSVF, another semi-autonomous behavior maps operator inputs for VNSVF navigation. The goal of this mapping is to decrease the user's mental load and increase comfort with VNSVF task completion. Since EEF motion is restricted to an array of precomputed poses, operator inputs can be simplified. The VF frame is a Task-Centered (TC) coordinate frame and was chosen for operator input mapping because TC frames

are intuitive, efficient, and easily understood [Hiatt and Simmons, 2006]. To provide this operator input mapping and VNSVF navigation a *RViz* plugin called Motion Command was developed. It was developed at NRG to have greater user interface flexibility compared to an Xbox controller or ROS interactive markers. Motion Command maps operator input from directionally descriptive buttons, labeled ‘Left’, ‘Right’, ‘Up’, ‘Down’, ‘In’, and ‘Out’, to movement around the VNSVF (Figure 3.5 right). In addition to mapping controls to intuitive buttons, the viewpoint in *RViz* follows the TC frame. The operator can change the viewpoint if desired and then return to the TC frame for the next motion. Thus, the directional button labels remain valid during task execution and reduce *context switching*.

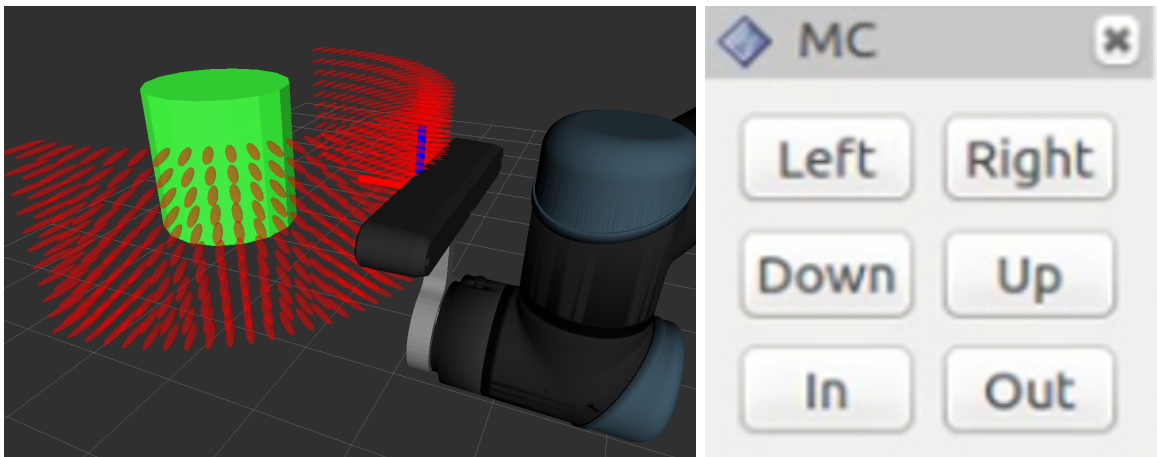


Figure 3.5: A VF layer (red markers) within a VNSVF around a cylindrical volumetric primitive FRVF (green cylinder) representing a storage canister. The *RViz* visualization also displays an ASUS Xtion Pro Live RGBD camera on a UR5 (lower right). Motion Command *RViz* plugin (bottom left) for TVF navigation [Sharp and Pryor, 2016].

VNSVFs were applied to both spatially discrete and spatially continuous domain tasks. Spatially discrete non-contact tasks can be executed with pose-to-pose control allowing the motion planning semi-autonomous behaviors the greatest flexibility with hardware constraints.

Examples include radiation surveys, which may require a counting time at each location, and visual inspection, where images can be stitched together if necessary. Spatially continuous tasks require surface interaction, and additional constraints must be placed on task execution. Some examples are painting, plasma cleaning, and laser cutting. Specific applications were manipulator to task transform determination, non-contact task execution, and D & D volume reduction. Two hardware platforms, a mobile manipulator and an industrial manipulator, were tested. Task evaluation required tool (URDF) and environmental (*MoveIt!* planning scene) information to check for collisions during motion planning.

3.2.1 Discrete Control for Non-Contact Task Execution

VNSVFs with position control allowed determination of the manipulator to the task surface transform for task completion. The VNSVF was placed at a 3D grid of locations in the manipulator’s virtual workspace. Each VNSVF pose is checked for an IK solution and then a motion plan to determine the locations reachability. The motion plan depends on the manipulator’s initial position and, therefore, this evaluation was hardware configuration and environmentally specific. The percentage of reachable VNSVF poses provides a quality metric for each manipulator to VNSVF transform. Metrics can be offline and displayed to aid the operator with transform selection. Then, the VNSVF and Motion Command provides operator assistance during task execution.

The manipulator to VNSVF transform experiment’s goal was to determine mobile manipulator base placement for an inspection task. The platform selected was the NRG’s ‘VaultBot’ which was co-developed with Clearpath Robotics (Figure 3.6). The 6DOF Universal Robotics UR5 industrial manipulators were chosen for their accuracy and robustness. They

also allow joint position, velocity, and torque motoring. Each UR5 has a payload of 5 kg, working radius of 850 mm, and a gravity compensated force following ‘teach’ mode. The Clearpath Husky required modifications to support dual UR5s including a steel mounting bulkhead and drive train upgrades. Modifications lowered the maximum speed to 0.5 m/sec and battery life to one hour. The VaultBot includes a sensor suite for navigation, Ubuntu laptop for ROS integration, and wireless router for autonomous functionality. Several LOAs are available such as single and dual manipulator *MoveIt!* control, joint control, EEF jogging through the controller, pose control with motion planning, and step-teleoperation [Sharp et al., 2017b]. *MoveIt!* control builds on (OMPL) [Sucan et al., 2012] which provides Probabilistic Road Maps and Rapidly-exploring Random Trees motion planning algorithms which require 6 DOF poses. The VaultBot was controlled from a base station through the wireless connection. RGBD camera data, image and point cloud, is displayed for the operator through the RViz interface.

The vault at Los Alamos National Laboratory (LANL) stores a variety of radioactive materials in a varied but limited set of containers (Figure 3.7). These materials must be periodically inventoried requiring significant personnel hours and radiological exposure. Limitations on allowed personnel time in the storage vault the tedious task nature resulted in NRG’s automated inventory proposal. Thus, the vault provided the initial motivation for the VaultBot’s experimental environment, and a virtual workspace of the lab setup at UT Austin was constructed. The environment included collision meshes of a storage container on a shelf located based on the VNSVF location in the test grid (Figure 3.8, left).

The grid of tested manipulator to VNSVF transforms was located to the VaultBot’s side due to a small manipulator workspace in front of the robot (Figure 3.8, right). The

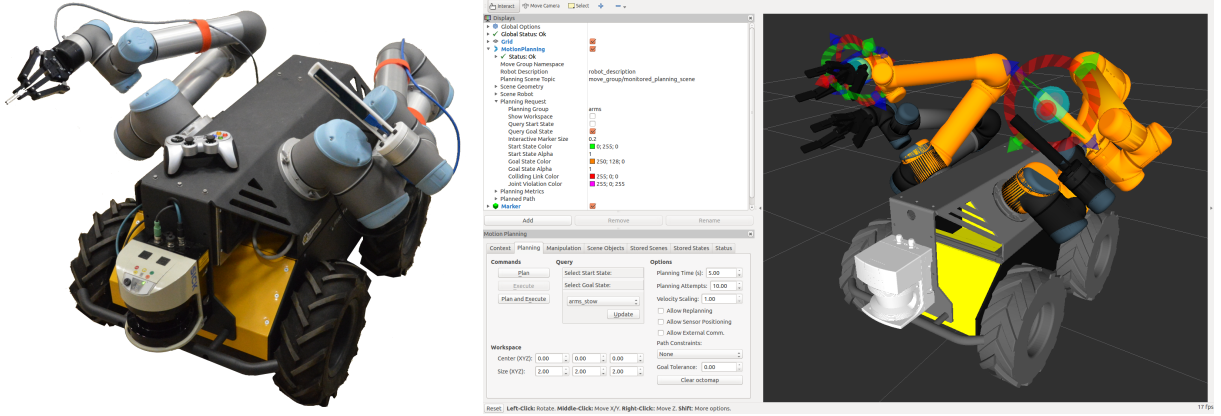


Figure 3.6: The NRG VaultBot mobile manipulator with attached Robotiq two finger gripper and RealSense R200 RGBD camera (left) and the VaultBot loaded into *RViz* with *MoveIt!* running. The orange version of the arms are the goal positions and can be altered with the 6 DOF red, green, and blue interactive markers (right).

testing grid was set to 5 cm resolution from 20 cm to 55 cm above the floor, -90 cm to 100 cm along the side of the VaultBot, and 60 cm to 120 cm out from the center of the VaultBot. The starting position for motion planning was the manipulator’s ‘home’ or storage position for mobile platform motion.

Testing the percentage of reachable VNSVF poses at each grid location provided a 3D heat map which was divided into horizontal layers for visualization (Figure 3.9). Additional heat maps are located in Appendix A. This approach allowed platform placement information displays based on VNSVF height. Appropriate locations were all grid points above a specified threshold. These locations were displayed for the operator as blue squares on the floor plane (Figure 3.8, left).

After base location determination, RGBD camera inspection using VNSVFs was demonstrated. Task objects were located and collision object loaded through AR fiducials,



Figure 3.7: The original storage can used for testing (left) and the Savy 4000 container being implemented at LANL (middle). Virtual workspace constructed from environmental information based on the lab setup at UT Austin (right).

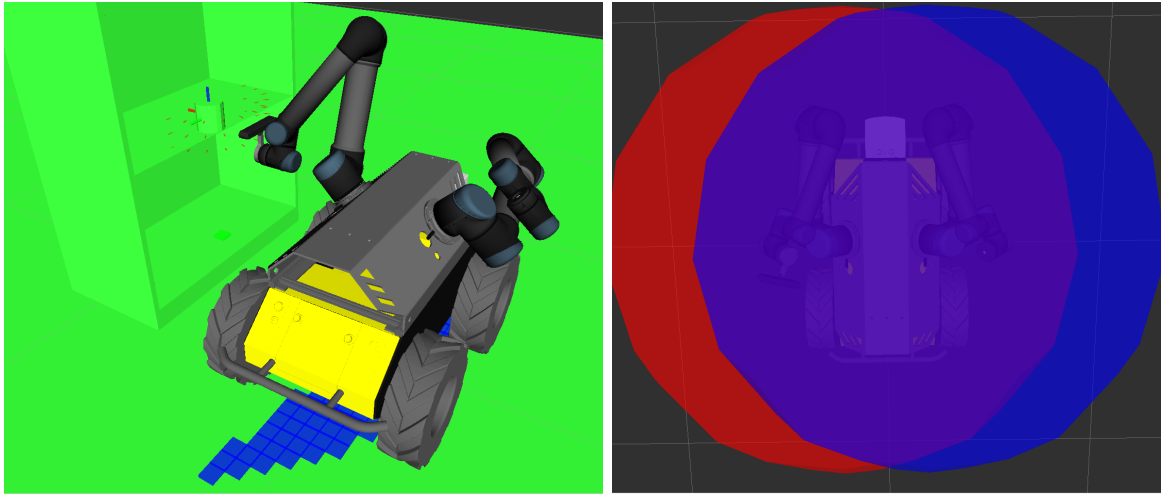


Figure 3.8: VaultBot visualization in *RViz* with environmental information (volumetric primitive, shelf, floor plane) and locations above the reachability percentage threshold marked in blue squares (left). VaultBot workspace (right) for the left UR5 (red) and right UR5 (blue).

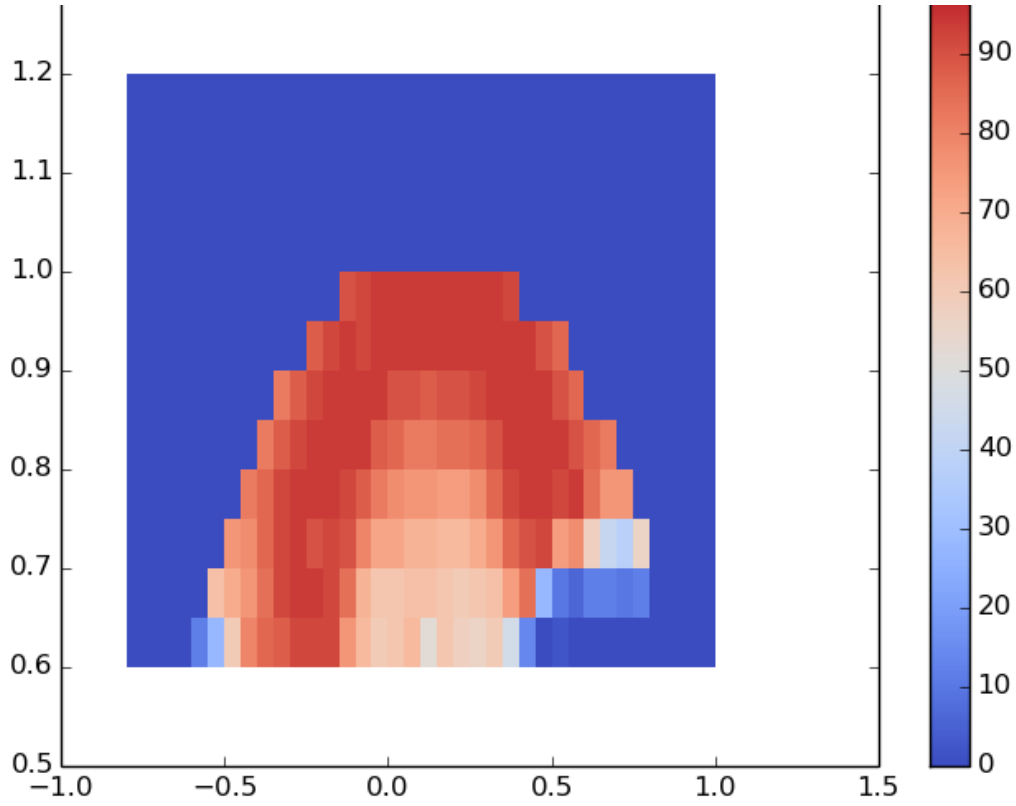


Figure 3.9: Mobile base placement for task execution results at 40 cm displayed as a heat map scaled in meters (right).

RGBD point cloud data, and the ROS package ‘ar_track_alvar’ [Scott Niekum, 2016]. Once located, motion to the VNSVF was planned and motion verified by the operator before execution. After entering the VNSVF, Motion Command buttons (Figure 3.5) change the camera position for canister inspection (Algorithm 2). After completing the inspection task, the left UR5 moves back to its home position and the VaultBot drives to the next task. The RGBD camera could be replaced with an alternative sensor, such as a radiation detector, while minimally affecting task execution.

Algorithm 2 Overall Task Flow

```
1: while system running do
2:   Get task frame pose by detecting and filtering ‘ar_track_alvar’ output
3:   Load surface and inspection information for detected fiducials
4:   Load FRVF collision mesh
5:   Calculate GVF based on loaded parameters
6:   Move EEF to TVF
7:   while inspection continues do
8:     Get operator input from Motion Command (Figure 3.5 buttons)
9:     if motion plan successful then
10:      Display plan for operator
11:      if operator approves then
12:        Execute motion
13:      else
14:        Ask for new operator input
15:      end if
16:    else
17:      Ask for alternate input
18:    end if
19:  end while
20:  Return manipulator to stow position
21: end while
22: Drive to next task
```

VNSVF assisted RGBD camera inspection was hardware demonstrated first with the storage container on a shelf and then with a pipe section. The first demonstration took place at UT Austin. The container was detected, localized, inspected by the operator, and the manipulator returned to ‘home’ position without collisions or requiring the operator to enter EEF poses (Figure 3.10).



Figure 3.10: The NRG VaultBot mobile dual manipulator *RViz* environment with shelf and object collision meshes. RGBD pointcloud data and Motion Command *RViz* plugin (bottom left) for TVF navigation are overlaid as the operator would see.

The second demonstration was the inspection of a horizontal pipe and was performed at the United States Department of Energy Portsmouth Gaseous Diffusion Plant during “EM Science of Safety: Robotics Challenge” [Pryor, 2017]. Task execution followed the same steps

as the previous demonstration. It also displayed the usefulness of VNSVFs extending the surface of interest instead of the entire volumetric primitive since several AR fiducials could be placed along a pipe allowing greater task flexibility.

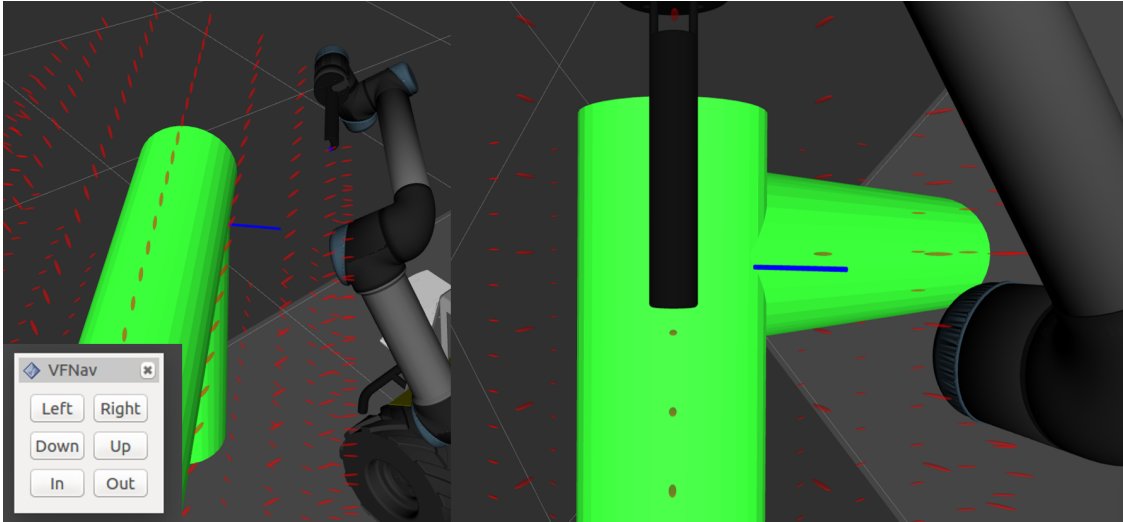


Figure 3.11: Overview of a pipe inspection task (left) and GVF maintained operator viewpoint for task execution (right).

3.2.2 Continuous Control for Non-Contact Task Execution

To also demonstrate VNSVFs on spatially continuous tasks, VNSVF construction and the Motion Command interface were expanded to include the *Descartes* path planning ROS package. The *Descartes* path-planning package for ROS provides a way to simultaneously plan for an entire trajectory instead of individual pose-to-pose moves. This allows *Descartes* to capitalize on semi-constrained trajectories to create favorable paths. It uses a graph-based search to find a low-cost trajectory given Cartesian points or Joint poses which appear more ‘comfortable’ to operators [Edwards, 2015b]. Additionally, *Descartes* enables users to specify

axis tolerances, such as allowing rotation about the tool for a welding torch or laser cutter and pitch of a circular cutting tool.

Once task information is provided, operators can utilize VNSVF poses for path construction. The current GVF pose can be added to the path by clicking the ‘Add Pt’ or ‘Remove Pt’ Motion Command buttons respectively (Figure 3.12, left). Once all desired VNSVF poses are added, clicking ‘Plan’ sends a pose array to *Descartes* to attempt planning a smooth trajectory. One shortcoming of *Descartes* is manipulator reconfigurations, such as flipping an elbow joint, may occur without warning the operator and thus motion plans are displayed. Operators approve plans by clicking ‘Execute’. Lower LOAs such as joint control and *MoveIt!* interactive markers are available during this process resulting in a variable autonomy system. The following demonstrations display instances where VNSVF generation techniques use geometric knowledge of task surfaces to augment D & D task execution. The industrial manipulator chosen for these demonstrations, due to previous experience with shorter reach models, was the Motoman SIA20 (Figure 3.12, left).

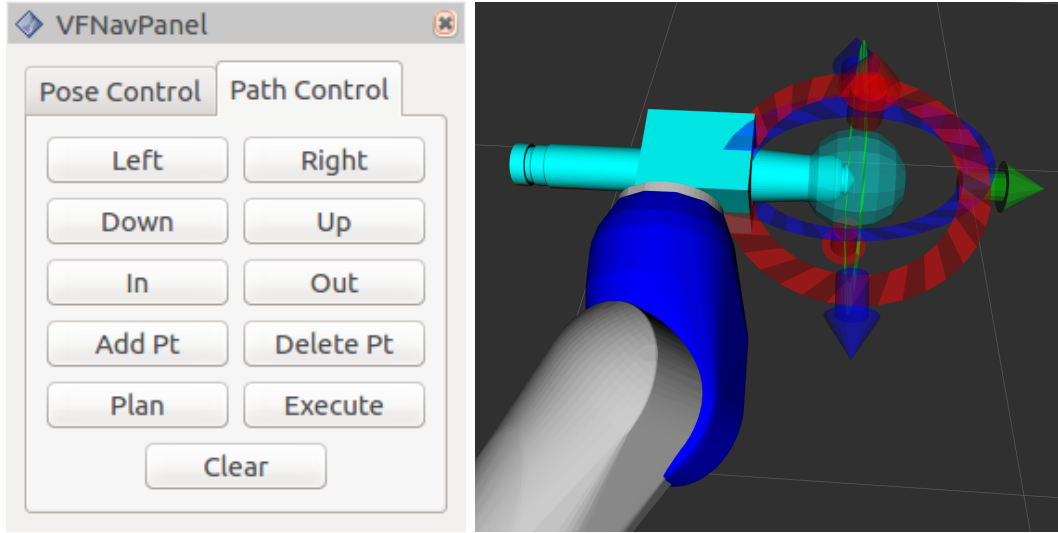


Figure 3.12: Motion Command *RViz* plugin for VNSVF navigation (left). SIA20 with a laser cutter tool in the *RViz* environment with 6 degrees of freedom interactive marker control (right).

Khan and Hilton discuss the benefits of using fiber optic laser systems to keep expensive laser generators away from contaminated materials (Figure 3.13). This process is a thermal size reduction/dismantling technique to reduce disposal costs. The laser cutter CAD model was designed based on an IPG Photonics Compact Cutting Head [Photonics, 2018]. It is approximately 344 mm long, 84 mm wide, and 84 mm deep (Figure 3.12, right). This laser cutter would have a slightly lower power capability at 4 kW compared to a 5 kW laser [Khan and Hilton, 2013; Hilton and Khan, 2014].

The least geometrically complex demonstration was a laser scabbling of a concrete section to remove the surface layer and contaminants. During scabbling, the laser head requires a specified surface offset while covering the entire surface evenly. Another demonstration is underwater laser cutting which also requires a constant standoff distance to ensure a dry zone

for optical surfaces. For tube reduction, the authors perform single or double linear passes with the focal position along the center of the tube (Figure 3.13). With this approach, there could be a lack of separation located at the side of the tube where both the standoff distance and material thickness were at maximums. A fourth demonstration consisted of passes where the system had an additional ‘side’ gas jet. The effect of the ‘side’ gas jet was to remove molten material from the cut. By lowering the cutting head during multiple passes, the laser was able to ‘gouge’ a deeper cut than normally achieved [Khan and Hilton, 2013; Hilton and Khan, 2014].

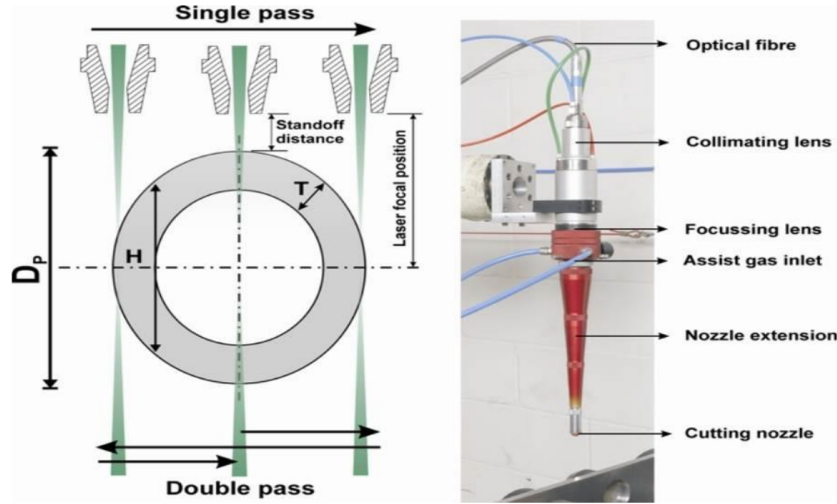


Figure 3.13: Process schematics and complete cutting head assembly [Khan and Hilton, 2013].

The first VNSVF assisted demonstration was planar laser scabbling with varied task parameters. Both simulations used $d_{min} = 10$ cm, $d_{max} = 30$ cm, and $d_{inter} = 5$ cm based on information from [Khan and Hilton, 2013]. The first trial’s d_{intra} was 20 cm (Figure 3.14, left) which left a limited number of GVF poses but a plan was still generated to cover the middle

of the planar surface. For the second trial d_{intra} was lowered to 10 cm. In this case, a longer path, covering the majority of the surface, was planned (Figure 3.14, right).

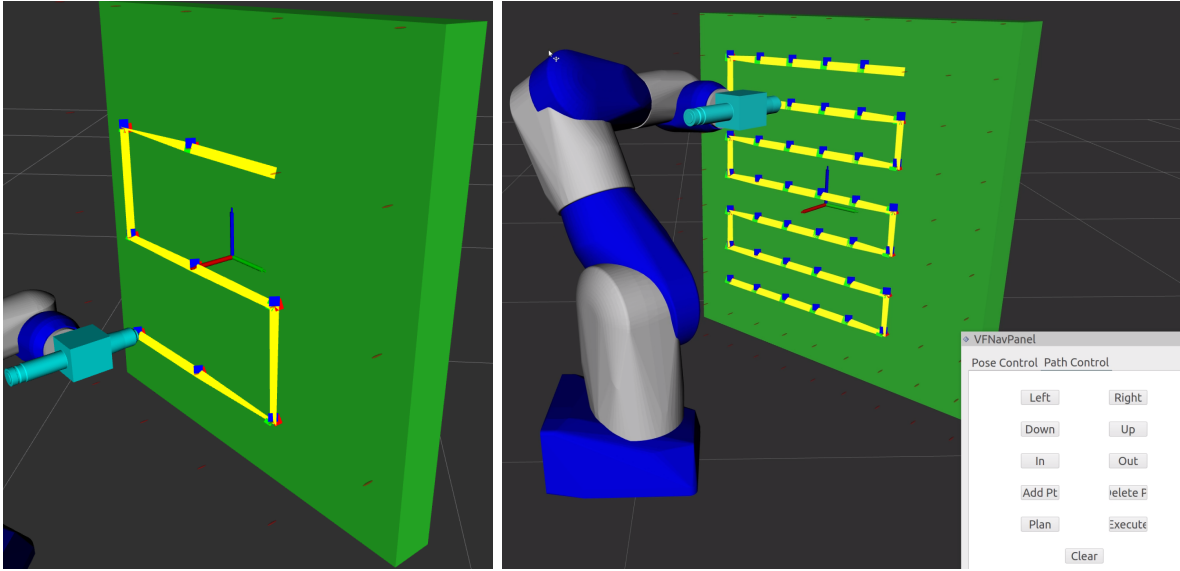


Figure 3.14: Demonstration showing planar shape primitive VNSVF poses (red markers) with 20 cm (left) and 10 cm (right) spacing and a planned path (yellow lines).

To further test planar VNSVF capabilities, a third trial, with $d_{intra} = 10$ cm used multiple GVF layers to simulate multiple passes of the laser cutter at decreasing standoff distances (Figure 3.15). This trial demonstrated the ‘gouge’ cutting technique [Hilton and Khan, 2014] and could apply to several of the selectively laser-cut samples [Khan and Hilton, 2013] such as the thick plates or concrete slab.

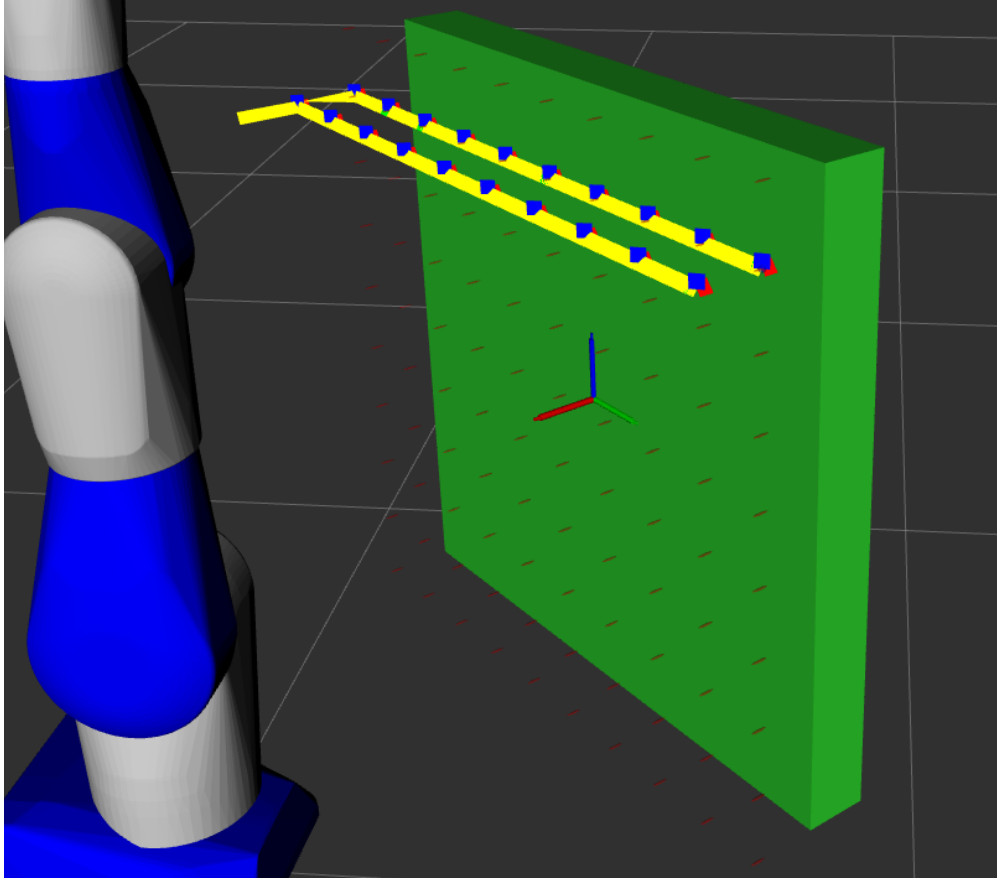


Figure 3.15: Demonstration showing planar shape primitive VNSVF poses (red markers) with 10 cm spacing and a planned multiple-pass ‘gouge’ cut (yellow lines) utilizing several GVF layers.

A 55 gallon drum, larger cylindrical primitive, has a diameter of 610 mm and a height of 880 mm. In this case, the SIA20 reach only covered approximately a quarter of the task surface (Figure 3.16). Similar workspace limitation would apply to the pressure vessel sample, $d = 600$ mm. Of particular interest in these cases is removing a task surface section to perform additional tasks on the interior. This approach would be similar to but more geometrically

complex than the “LaserSnake” mock pressure vessel demonstration [Khan and Hilton, 2013]. Another possible demonstration would be to merge ‘gouge’ cutting with curvature following to increase cutting depths for thick-walled vessels.

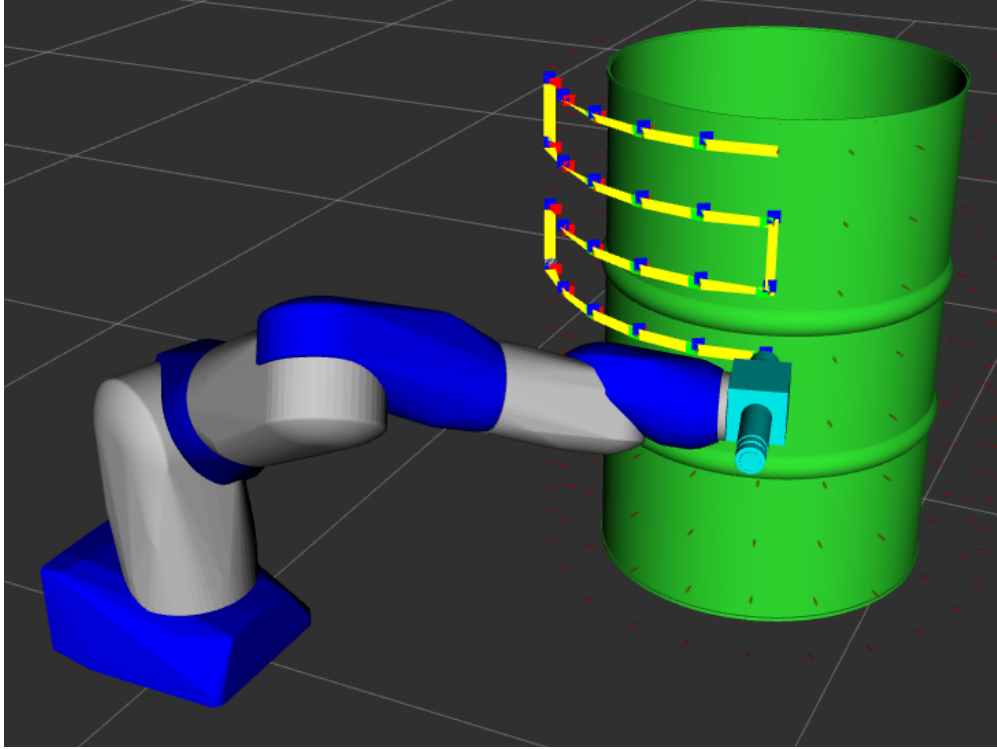


Figure 3.16: Demonstration showing cylindrical shape primitive VNSVF poses (red markers) with 10 cm spacing and a planned path (yellow lines).

The “nuclear jungle” motivated the complex ‘hybrid’ environment which contains multiple types of shape primitives (Fig. 3.17). There is a wall, several pipes ($d = 60$ mm, $l = 2$ m), and a 55 gallon drum in the environment (Fig. 3.18). One demonstration displays a planned path around one of the pipes (Fig. 3.18). Given environmental constraints, selectively dismantling of the front pipes to cut the back pipes would be required similarly to [Khan

and Hilton, 2013]. Another demonstration displays wall coverage while avoiding emerging piping (Fig. 3.19).



Figure 3.17: “Nuclear Jungle” tube cutting demonstration before (left) and after (right) from [Khan and Hilton, 2013].

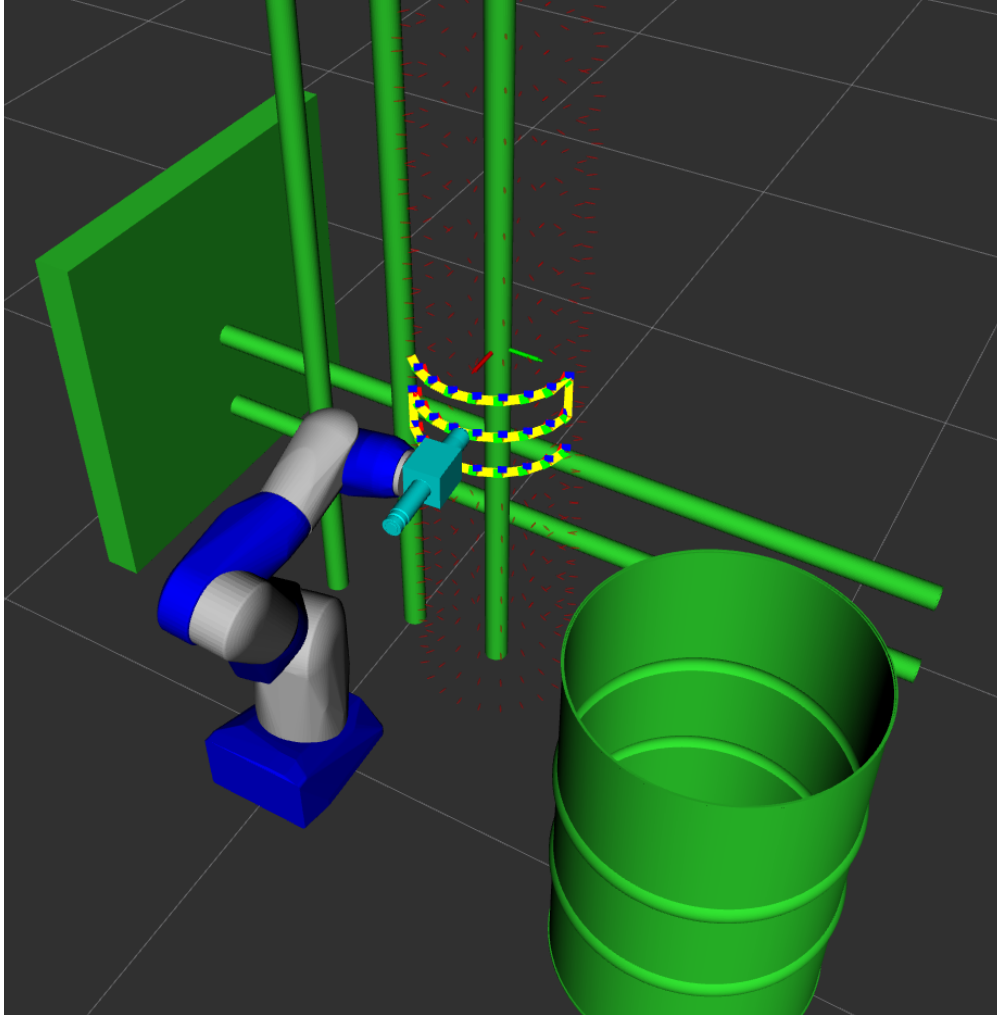


Figure 3.18: Demonstration of the “nuclear jungle” with cylindrical shape primitive VNSVF poses (red markers) with 10 cm spacing around 60 mm piping and the planned paths (yellow lines).

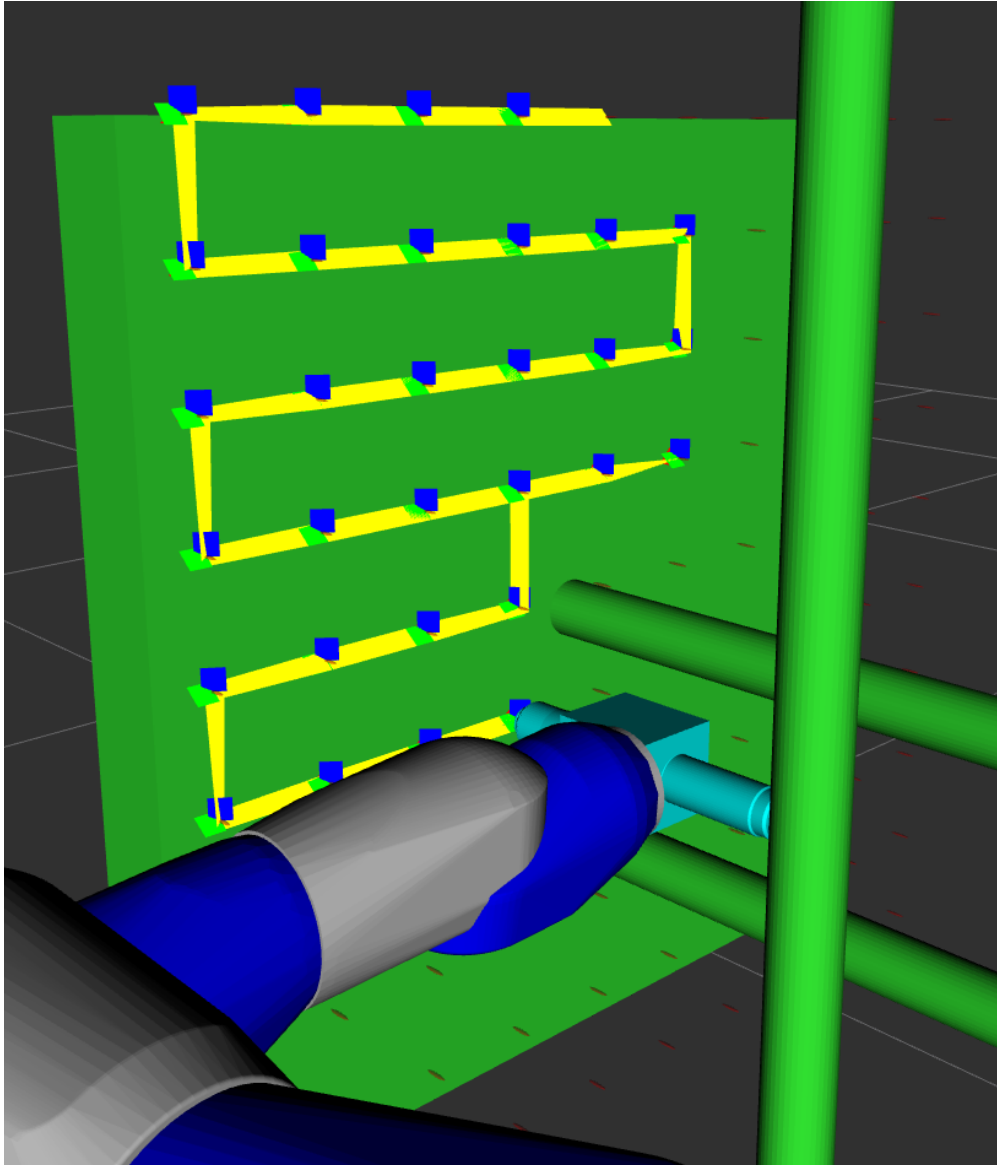


Figure 3.19: Demonstration of the “nuclear jungle” with planar shape primitive VNSVF poses (red markers) with 10 cm spacing with the planned path avoiding piping (yellow lines).

3.3 Volumetric Primitive Limitations

The combination of a task-defined polygonal mesh FRVF and point cloud GVF layers into a TVF is more complete VF representation than previous methods. Many tasks, both spatially discrete and spatially continuous, can be represented by a volumetric primitive. Thus, preliminary testing with VNSVFs was performed to determine TVF viability. Demonstrations included manipulator to task surface transform selection for a mobile manipulator, non-contact inspection of a storage container on a shelf & section of pipe, and path generation for D & D volume reduction. Testing with mobile manipulator base placement was successful in providing operators with base locations for task completion. Two spatially discrete RGBD inspections tasks were demonstrated, on hardware, where operators used the VNSVF to approach the task surface, complete the task, and move away from the task surface. D & D applications, which are spatially continuous instead of spatially discrete, were also demonstrated in multiple simulations. The *Descartes* ROS package’s use of semi-constrained paths allows trajectory optimization if a manipulator degree of freedom is unconstrained like with laser cutting where the tool may freely rotate around the cutting beam. The VNSVF demonstrations showed the feasibility and potential usefulness of using VNSVFs to augment several types of non-contact tasks on multiple hardware platforms.

The ability to select from layers of precalculated poses, already properly located and orientated for the task, reduces the mental burden on the operator. However, not all tasks, spatially discrete or spatially continuous, can be simplified to a volumetric primitive. These include non-linear facility piping infrastructure, abandoned machinery, and individual complex components. Therefore, there are significant shortcomings in the VNSVF approach. While VNSVFs are more descriptive than previous approaches, they lack the ability to fulfill

several needs expressed in the literature. These requirements were outlined in the summary by [Bowyer et al., 2014], “more **complete representation** of the working environment”, and [Hilton and Khan, 2014], “a highly automated remote technology that can deliver a non-contact smarter dismantling process, cut most materials, cut **complicated structural geometries...**”. As such, to achieve broad applicability for the tasks and domains identified, TVF principles must be extended to complex task surface geometries.

Chapter 4

Complex Geometry Task Virtual Fixture Development

This effort’s primary focus is to develop more expressive VFs by expanding upon current VF generation methods¹. Therefore, TVF principles were applied to VNSVFs, which are acceptable for some tasks. Limitations and lessons learned during VNSVF development aided in the development of complex surface geometry TVFs. For the rest of this document Task Virtual Fixture (TVF) refer to complex task geometry TVF. TVF development is the core of this effort’s contribution.

Many non-contact tasks require a specific tool to task surface transform, [Khan and Hilton, 2013], and the normal vector, $\hat{\mathbf{n}}$, at a task surface location (Equation 4.1) acts as a ‘depth’ direction. This *a priori* knowledge allows the removal of some transform calculations from the operator’s *context switching* mental load. EEF motion can, therefore, be restricted to a GVF offset in 3D space from the task surface [Sharp et al., 2018]. Multiple GVF layers and the ability for operators to choose from precalculated poses, instead of having to teleoperate without specific location information, maximizes TVF usefulness. As with VNSVFs, TVF layers are parallel to the task surface to limit interaction, unlike other previous implementations [DeJong et al., 2006]. TVFs also do not restrict other LOAs similarly to previous approaches [Schroeder and Pryor, 2011; Sharp et al., 2017b].

¹This chapter includes material published in Sharp and Pryor [2018] where I contributed that form of TVF generation pipeline, the object testing, and the analysis of the results.

$$\begin{aligned}
P_{surf}(i).v &= \begin{bmatrix} v_x & v_y & v_z \end{bmatrix} \\
P_{surf}(i).\hat{\mathbf{n}} &= \begin{bmatrix} n_x & n_y & n_z \end{bmatrix}
\end{aligned} \tag{4.1}$$

TVFs are generated via a series of algorithms (Figure 4.1) where the inputs are a task surface and task parameters. VF surfaces are calculated based on minimum & maximum distances (d_{min} , d_{max}) and intralayer & interlayer distances (d_{intra} , d_{inter}) task parameters. The interlayer distance, d_{inter} , defines the distance between VF layers while intralayer resolution, d_{intra} , determines the maximum distance between points within a VF layer. TVFs can be created and stored for multiple tasks on a single surface or *vice versa*.

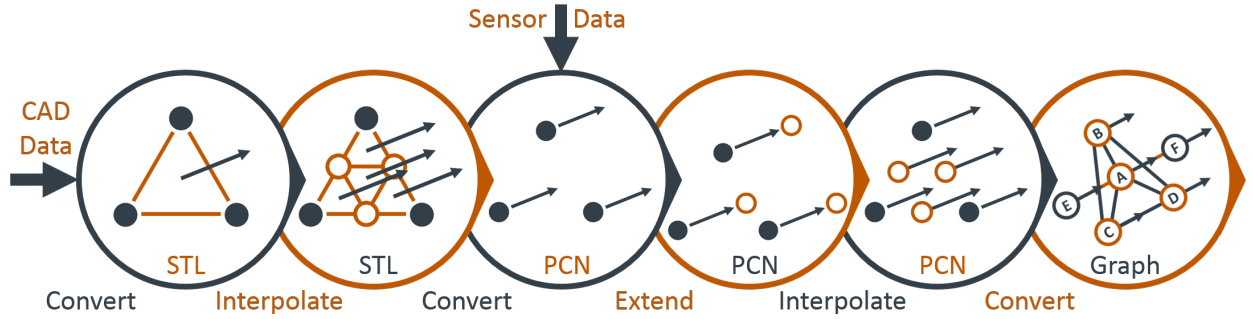


Figure 4.1: TVF generation algorithms to go from a polygonal mesh or sensor data to a bi-directional graph structure.

4.1 Task Surface Pipeline Input

Once the required task parameters have been defined the task surface must be provided. During this effort's early stages the question of pipeline input file format was considered since it will affect later algorithms. The two major data formats are spline based which include

the majority of CAD software output and polygonal meshes. Spline-based models, also known as Non-uniform Rational Basis Splines (NURBS), are precise mathematical surface representations created by calculating the tensor product of two NURBS curves (Figure 4.2, left). NURBS is a popular representation in CAD, CAM, and CAE software packages [PTC, 2017; SOLIDWORKS, 2017] and the most generic file types are STEP and IGES. Polygonal meshes store data as polygon vertices and surface normal (Listing 4.1). A polygonal mesh's complexity is dependent on its instantiative methodology and the level of precision can be chosen based on task parameters (Figure 4.2, middle and right). The most common polygonal file formats are STL, DAE, and PLY.

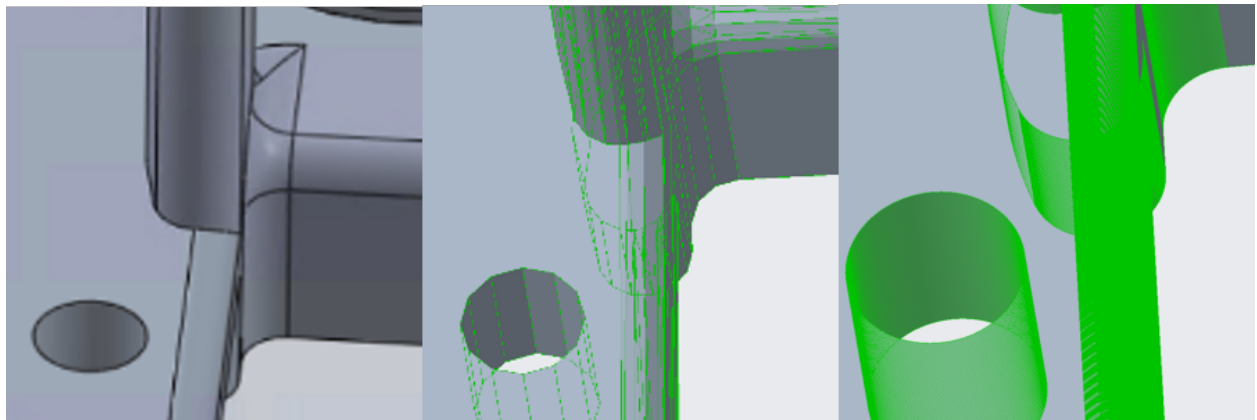


Figure 4.2: Spline based surface (left), low detail STL (middle), and high detail STL (right) [PTC, 2017].

Code 4.1: Triangular Polygonal Mesh File Format

```
facet normal nx ny nz
  outer loop
    vertex v1x v1y v1z
    vertex v2x v2y v2z
    vertex v3x v3y v3z
```

```
        endloop
    endfacet
```

Among the justifications for polygonal mesh data formats is the difficulty in converting from polygonal meshes to spline based models. Surface reconstruction methods are well researched but likely require operator input to achieve task acceptable meshes [Botsch et al., 2010], [Innvometric, 2018c], [PTC, 2017]. This process is similar to fitting curves to 2D data where data point choice has high importance. The process in 3D increases in complexity and requires human assignment of fitting points, possibly with repeated attempts, to convert the model from a polygonal mesh to a spline based model. For example, users must fill in holes in the surface (Figure 4.3, left), place curves at surface boundaries, and place a series of control points onto the surface (Figure 4.3, middle). Once a surface section is enclosed, by control points, the program fits a 3D curve to surface patches (Figure 4.3, right). This process was demonstrated using PolyWorks [Innvometric, 2018c] but other programs follow a similar methodology [PTC, 2017]. Surface conversion time is dependent on operator experience level and model complexity. Therefore, this is a difficult process to automate which would require operator intervention when constructing TVFs from sensor data.

Converting the other direction, from spline based models to polygonal meshes is completed using curve sampling (Figure 4.2, left). Using polygonal meshes as pipeline input does not prevent spline based file integration, but does add a conversion step. Down-conversion can also be accomplished while retaining information based on task requirements (Figure 4.2, middle and right). Models can also be drawn from online non-spline databases such as Thingiverse [Thingiverse, 2018] and 3DWarehouse [SketchUp, 2017a,b] which contain millions of polygonal meshes.

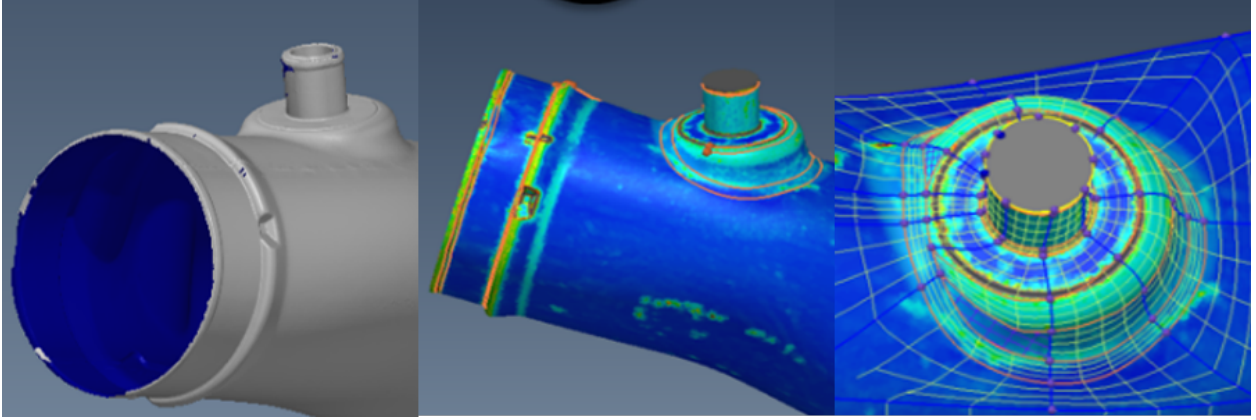


Figure 4.3: Filling in polygonal mesh holes (left), placing control points, orange points and lines (middle), and conversion to NURBS patches, blue and yellow lines (right) in PolyWorks Modeler [Innvometric, 2018b].

Another supporting reason to originate TVF generation with polygonal meshes is sensor data integration. Polygonal meshes can be created from sensor data where detail is determined by sensor resolution (Figure 4.4). Sensor data input would add a pre-processing step but conversion into a polygonal mesh is much less strenuous than conversion to a spline based model. Possible sensors for surface data gathering include both inexpensive RGBD sensors like an Asus Xtion [ASUS, 2015] and metrology grade non-contact scanners such as the FARO Cobalt [FARO, 2017]. Metrology scanners are progressing to the point where they obtain <0.05 mm accuracy and <0.1 mm point spacing [FARO, 2018]. In these cases, the algorithms possess benefits for “open world” scenarios and thus positively impact a wide range of robot applications including emergency response and remote exploration. Future work may investigate extending TVF generation to automate additional file type inputs.

In order to utilize spline based models, polygonal meshes, and sensor data triangular polygonal meshes were chosen as the input for the TVF generation pipeline. The stored data

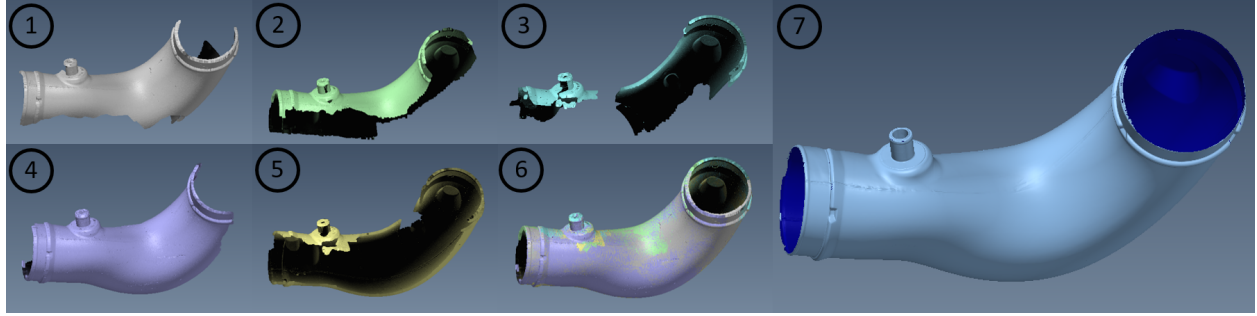


Figure 4.4: Point cloud data merging (1-5 into 6) and conversion into a polygonal mesh (7) in PolyWorks Inspector [Innvometric, 2018a].



Figure 4.5: FARO Cobalt 9MP (left) and Faro Cobalt mounted to a Universal Robotics UR3 on a granite measurement table (right).

is divided into triangles $T(i)$ where $i = 1, \dots, N$ and N is the number of triangles (Listing 4.1). Each triangle includes its surface normal and three vertices (Equation 4.2).

$$\begin{aligned}
 T(i).\hat{\mathbf{n}} &= \begin{bmatrix} n_x & n_y & n_z \end{bmatrix} \\
 T(i).v(j) &= \begin{bmatrix} v(j)_x & v(j)_y & v(j)_z \end{bmatrix} \\
 j &= 1, 2, 3
 \end{aligned} \tag{4.2}$$

4.2 Polygonal Mesh Verification and Interpolation

The Google 3D Warehouse [SketchUp, 2017a] (Figure 4.6) is an online database of thousands of *SketchUp* models [SketchUp, 2017b] (Figure 4.7) which can be downloaded in the polygonal mesh DAE format. Integrating this large body of polygonal mesh models would be beneficial for thorough and unbiased system evaluation.

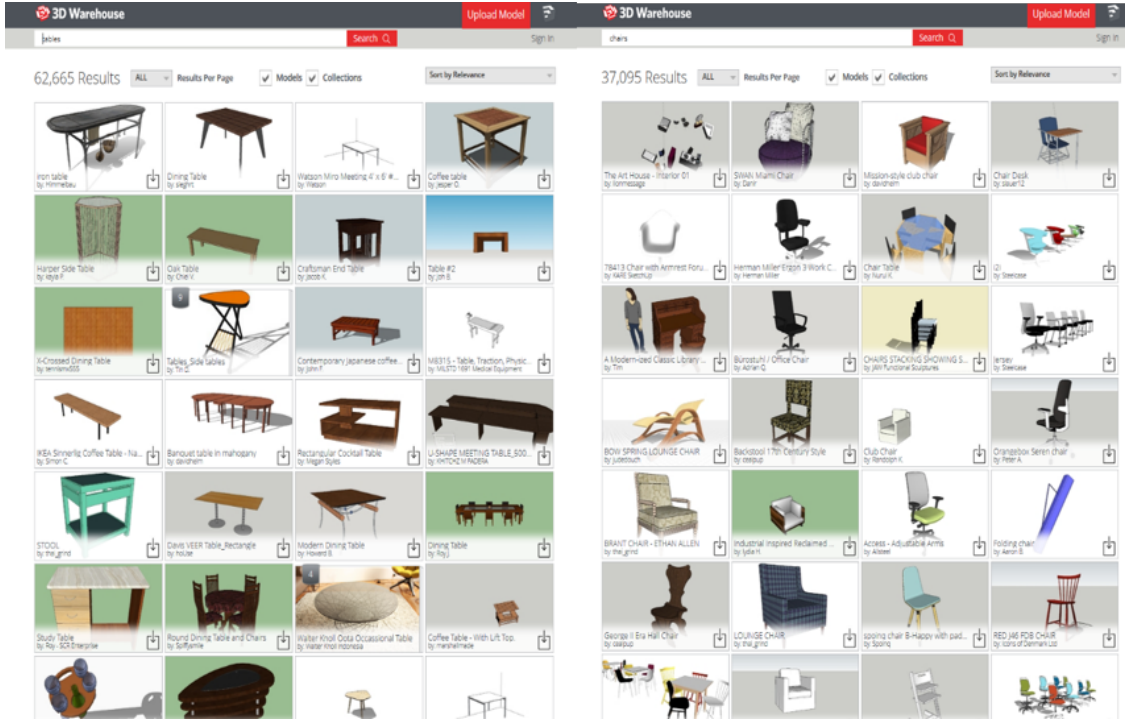


Figure 4.6: Screenshots of the searches of ‘table’ (left) and ‘chair’ (right) of Google 3D warehouse [SketchUp, 2017a].

However, preliminary investigations found Google 3D Warehouse and *SketchUp* files commonly contain another surface defect. Models may store two triangles with the same Cartesian vertex coordinates but opposite surface normals (Equation 4.3). Thus, $T_{surf}(i) \cdot \hat{\mathbf{n}}$

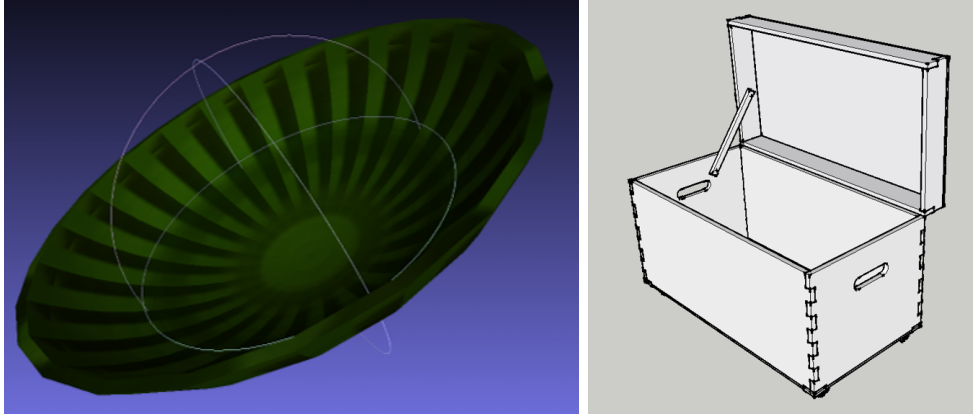


Figure 4.7: Box (left) and bowl (right) *SketchUp* [SketchUp, 2017b] models.

are pointed both out of and into the surface for each polygon. Therefore, the mesh is non-manifold.

$$\begin{aligned}
 T_{surf}(i).v &= T_{surf}(j).v \\
 T_{surf}(i).\hat{\mathbf{n}} &= -T_{surf}(j).\hat{\mathbf{n}}
 \end{aligned}
 \tag{4.3}$$

Automating a $T_{surf}(i).\hat{\mathbf{n}}$ correction process is outside this effort's scope but a straightforward check for duplicate surface triangles was implemented. $T_{surf}(i).\hat{\mathbf{n}}$ are area weighted and averaged over the entire mesh (Algorithm 3). As with a sphere, this calculation yields a near zero value for quality meshes since all polygons approximately cancel out. However, if mesh regions are inconsistent, the average $\hat{\mathbf{n}}$ will be skewed in one direction. Therefore, if the magnitude of the average $\hat{\mathbf{n}}$ is higher than a threshold, ε_1 , the point's mesh section is likely to possess defects. When point normal calculations are complete for the entire mesh, the percentage of points with warnings is output to the operator where high percentages indicate

defective meshes (Algorithm 5). 3D Warehouse models can be converted to a acceptable polygonal mesh format through the paid version, *SketchUp* Pro. If inconsistencies are detected the mesh should be reconstructed using any of numerous techniques [Botsch et al., 2010] prior to use as input to the TVF generation pipeline.

Algorithm 3 Mesh Normal Calculation

```

1:  $\mathbf{n}_{\text{sum}} = \begin{bmatrix} 0_x & 0_y & 0_z \end{bmatrix}$ ,  $num_{tri} = 0$ ,  $sum_A$ 
2: for  $T_{surf}(j) : j = 1, \dots, J$  where  $J$  is the total triangles do
3:    $\mathbf{n}_{\text{sum}} + = T_{surf}(j) \cdot \hat{\mathbf{n}} * T_{surf}(j)_A$ 
4:    $num_{tri} + = 1$ ,  $sum_A + = T_{surf}(j)_A$ 
5: end for
6:  $\bar{\mathbf{n}}_{\text{sum}} = \frac{\mathbf{n}_{\text{sum}}}{num_{tri} * sum_A}$ 
7: if  $|\bar{\mathbf{n}}_{\text{sum}}| > \varepsilon_1$  then
8:   Report warning to operator.
9: end if

```

Once the task mesh, T_{surf} , is provided and checked, the triangle side lengths are compared to the task intralayer distance, d_{intra} . Triangles with sides exceeding d_{intra} are subdivided iteratively until appropriately sized (Algorithm 4). Points placed on each side midpoint and divide the triangle into four triangles geometrically similar to the original triangle (Figure 4.8). This primal approximating dissection method was chosen to maintain original surface points, retain fine surface features, and increase point dispersion over simpler triangle bisection method. More complex methods for mesh subdivision are available including parameterization-based or surface-oriented remeshing techniques [Botsch et al., 2010] and non-linear subdivision [Aspert et al., 2003], [Schaefer et al., 2008].

Algorithm 4 Triangle Interpolation

```
1: for  $T(i) : i = 1, \dots, N$  where  $N$  is the total triangles do
2:    $d_{12} = \sqrt{(v1_x - v2_x)^2 + (v1_y - v2_y)^2 + (v1_z - v2_z)^2}$ 
3:    $d_{23} = \sqrt{(v2_x - v3_x)^2 + (v2_y - v3_y)^2 + (v2_z - v3_z)^2}$ 
4:    $d_{31} = \sqrt{(v3_x - v1_x)^2 + (v3_y - v1_y)^2 + (v3_z - v1_z)^2}$ 
5:   if  $d_{12}$  or  $d_{12}$  or  $d_{12} > d_{intra}$  then
6:      $mid_{12} = \begin{bmatrix} (v1_x + v2_x)/2, \\ (v1_y + v2_y)/2, \\ (v1_z + v2_z)/2 \end{bmatrix}$ 
7:      $mid_{23} = \begin{bmatrix} (v2_x + v3_x)/2, \\ (v2_y + v3_y)/2, \\ (v2_z + v3_z)/2 \end{bmatrix}$ 
8:      $mid_{31} = \begin{bmatrix} (v3_x + v1_x)/2, \\ (v3_y + v1_y)/2, \\ (v3_z + v1_z)/2 \end{bmatrix}$ 
9:      $T(i).v = [v1, mid_{12}, mid_{31}]$ 
10:     $T_{new1}.v = [v2, mid_{23}, mid_{12}]$ 
11:     $T_{new2}.v = [v3, mid_{31}, mid_{23}]$ 
12:     $T_{new3}.v = [mid_{12}, mid_{23}, mid_{31}]$ 
13:     $T_{new1}.\hat{n} = T_{new2}.\hat{n} = T_{new3}.\hat{n} = T(i).\hat{n}$ 
14:    Add new triangles to all triangles
15:   end if
16: end for
```

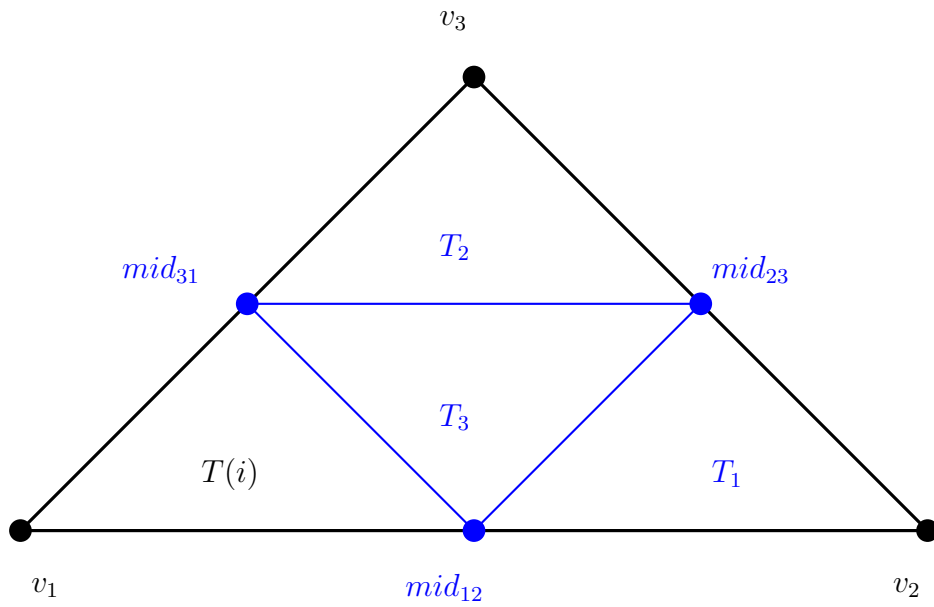


Figure 4.8: Visualization of the triangle interpolation process from Algorithm 4. Original points and sides are in black. Added points and sides are blue.

4.3 Surface Normal Calculations

Point normals (Equation 4.1) are calculated with a normalized weighted average of the triangle's $\hat{\mathbf{n}}$ for triangles in which the point is a vertex similarly to [Xiuzhi Qu and Brent Stucker, 2003]. Each triangle normal is weighted by the ratio of the triangle's area to the area of all triangles in which the point is co-located (Algorithm 5). The integration of triangle area weighting better accommodates triangle size variations than averaging. Thus, the Delaunay neighborhood utilized similarly to [Ma et al., 2013], but the polygonal mesh file provides normal information.

Algorithm 5 Point Normal Calculation

```

1: warnings = 0, percentage = 0
2: for  $P_{surf}(i); i = 1, \dots, I$  where  $I$  is the total points do
3:    $\mathbf{n}_{sum} = [0_x \ 0_y \ 0_z]$ ,  $num_{tri} = 0$ ,  $sum_A$ 
4:   for  $T_{surf}(j) : j = 1, \dots, J$  where  $J$  is the total triangles do
5:     if  $P_{surf}(i).v \in T_{surf}(j).v$  then
6:        $\mathbf{n}_{sum} += T_{surf}(j).\hat{\mathbf{n}} * T_{surf}(j)_A$ 
7:        $num_{tri} += 1$ ,  $sum_A += T_{surf}(j)_A$ 
8:     end if
9:   end for
10:   $\bar{\mathbf{n}}_{sum} = \frac{\mathbf{n}_{sum}}{num_{tri} * sum_A}$ 
11:  if  $|\bar{\mathbf{n}}_{sum}| < \varepsilon_2$  then
12:    warnings = warnings + 1
13:  end if
14:   $P_{surf}(i).\hat{\mathbf{n}} = \frac{\bar{\mathbf{n}}_{sum}}{|\bar{\mathbf{n}}_{sum}|}$ 
15: end for
16: percentage =  $\frac{warnings}{I}$ , Report percentage of warnings to operator.
```

Taking advantage of polygonal mesh information also avoids recalculating surface information with point neighborhood Plane Fitting (PF) approaches. The neighborhood may be determined with a $k_{nearest}$ neighbor of r_{radius} neighborhood calculation. The Least Squares

Method PF solution is the eigenvector corresponding to the smallest eigenvalue which is found by analyzing the eigenvalues and eigenvectors of the covariance matrix (Equation 4.4). Since both vectors perpendicular to the calculated plane, out of and into the surface, are mathematically valid, this method requires additional information such as a sensor viewpoint to properly orient point normals [Rusu, 2009]. In the case of an entire model, a single viewpoint is unable to have line of sight to the full task surface. This results in model self-occlusions and thus in incorrect point normals for occluded mesh regions.

$$\begin{aligned}
\bar{X} &= \frac{\sum_{i=1}^n x_i}{n} \\
\bar{Y} &= \frac{\sum_{i=1}^n y_i}{n} \\
m &= \frac{\sum_{i=1}^n (x_i - \bar{X})(y_i - \bar{Y})}{\sum_{i=1}^n (x_i - \bar{X})^2}
\end{aligned} \tag{4.4}$$

Consider a surface with two raised portions (Figure 4.9). At $k = 3$ the surface normals, calculated with Least Squares Method (Equation 4.4) for each point include only the point of either side of the one in question resulting in the angled surface normals at the corners. As the value of k is increased surface normals smoothing continues. At higher values for k the all surface normals approach the average normal for the surface (Figure 4.9).

PF algorithms function best with point distribution consistency [Ma et al., 2013], which may be lacking in the surface mesh [Sharp and Pryor, 2018]. The effect of non-uniform point distribution is shown on superellipsoid surfaces for $k_{nearest}$ neighbor (Figure 4.10,

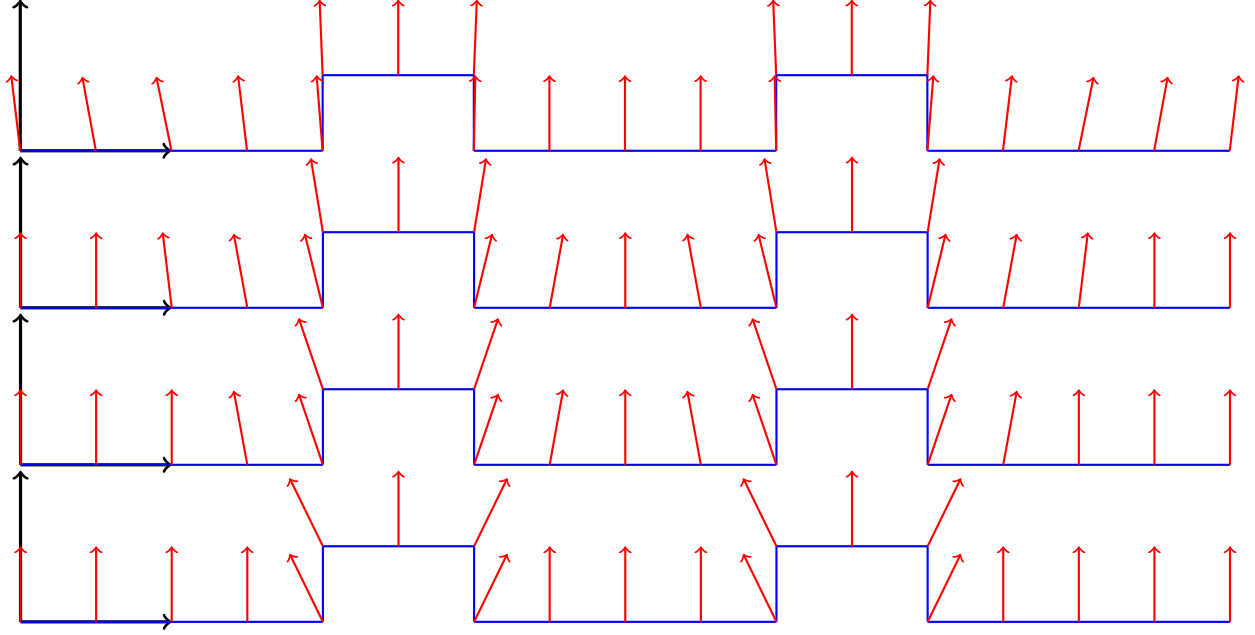


Figure 4.9: Example of the effect of variation of k nearest neighbors for normal vector calculations on small features. The values of k increase from bottom to top (3, 5, 7, 11). Numerical values were calculated through Least Squares Method (Equation 4.4) and are presented in Appendix B.

top), r_{radius} neighborhood (Figure 4.10, middle), and the triangle normal averaging chosen (Algorithm 5) (Figure 4.10, bottom). If future algorithm development requires additional surface information, such as curvature, PF method [Zhihong et al., 2011; Foorginejad and Khalili, 2014] integration into the TVF generation pipeline is an option.

During preliminary investigations, defects in the input surface meshes were discovered [Sharp and Pryor, 2018]. Inverted $T_{surf}(i) \cdot \hat{\mathbf{n}}$ regions can affect VF layer generation. Thus, a second straightforward check was implemented to warn operators of regions with $T_{surf}(i) \cdot \hat{\mathbf{n}}$ regions was implemented. Once the area weighted summation is complete and prior to re-normalization, the magnitude of the vector is calculated and compared to a threshold. If

the vector magnitude is less than ε_2 , a small value, the point’s mesh section is likely to possess defects (Algorithm 5). When point normal calculations are complete for the entire mesh, the percentage of points with warnings is output to the operator where high percentages indicate defective meshes. Mesh checks were implemented to warn operators, but defective meshes are expected to be uncommon, particularly, when professional software is used to generate the mesh [PTC, 2017; SOLIDWORKS, 2017; Innvometric, 2018c]. Such defects also cause problems 3D printing models. Since sharing models to 3D print is one of the main goals of several online repositories [Thingiverse, 2018], unprintable meshes are unlikely to be shared.

Once polygonal mesh checks, interpolation, and conversion to a point cloud with normals are complete, the task surface cloud with normals is voxelized. Voxel filtering discretizes space into boxes of a provided size and all points present within the same box are approximated a point located at their centroid [Rusu, 2009]. To maintain high surface density but be sure to remove duplicate points, the discretization box is a cube with side lengths which are a small percentage of the task d_{intra} .



Figure 4.10: Superellipsoid example of $k_{nearest}$ neighbor (5 top left, 20 top right) and r_{radius} neighborhood ($r_{radius} = d_{intra}$ middle left, $r_{radius} = 5 * d_{intra}$ middle right) surface normal estimation. Example of d_{intra} effects on STL interpolation and surface normal calculation (0.5 m) bottom left, (0.5 m) bottom right). Viewed with PCLVisualizer [PCL, 2018].

4.4 Virtual Fixture Layer Construction

VF offset surfaces are calculated from a task surface's discrete surface information, $P_{surf}(i)$ (Equation 4.1), and task parameters. VF layers are constructed between minimum, d_{min} , and maximum, d_{max} , at intervals of d_{inter} (Equation 4.5). The extreme distances are enforced by a lack of VF layers outside of the specified parameters to eliminate unintended interactions. Cases where d_{max} is greater than an interior space dimension must also be considered. Using a task specified surface as an FRVF handles these cases since the VF layer region would be located within a forbidden region. Complications can arise as $P_{surf}(i).\hat{\mathbf{n}}$ are extended away from the task surface due to normal vector density changes caused by convergence or divergence phenomena (Figure 4.11).

$$\begin{aligned}
 d_{layer} &= d_{min} + d_{inter} * layer, layer = 0, \dots, k \\
 P_{vf}(i).v &= T_{surf}(i).v + T_{surf}(i).\hat{\mathbf{n}} * d_{layer} \\
 P_{frvf}(i).\hat{\mathbf{n}} &= T_{surf}(i).\hat{\mathbf{n}}
 \end{aligned} \tag{4.5}$$

$$P_{gvf}(i).\hat{\mathbf{n}} = -T_{surf}(i).\hat{\mathbf{n}} = \begin{bmatrix} -n_x & -n_y & -n_z \end{bmatrix}$$

Convex regions, \mathbb{R}_{vex} , have diverging surface normals, $\forall P_{surf}.\hat{\mathbf{n}} \in \mathbb{R}_{vex}$, resulting in decreasing $\hat{\mathbf{n}}$ densities. Therefore, intersections will not be present (Figure 4.11, outer) but interpolated points may be required to maintain VF intralayer resolution. After extending $T_{surf}(i).\hat{\mathbf{n}}$, previous spatial information is unreliable. Therefore, point neighborhood information is used for VF layer interpolation since spatially organized information is absent. To limit extraneous point creation which contributed to extended computation times, a minimum interpolation radius distance, r_{min} , was implemented (Figure 4.12). This update required

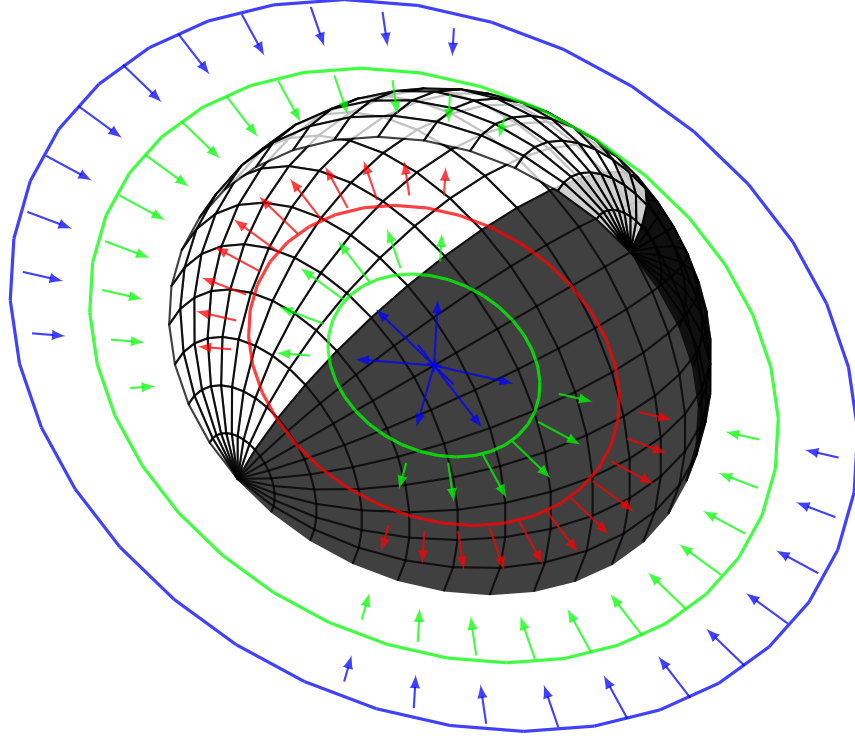


Figure 4.11: Visualization of a hemispherical task surface and VF normal vectors (Equation 4.5). Two VF layers are displayed on the convex outside and three VF layers on the concave inside. The number of vectors decreases with distance on the concave interior.

neighborhood calculations to change from a $k_{nearest}$ neighborhood to a r_{radius} neighborhood. The algorithmic alteration also allows r_{radius} to be changed proportionally to d_{layer} , r_{prop} . Thus, an approximate percentage of the VF layer surface area is maintained with increasing VF layer distance while avoiding interpolating through thin surfaces. Distance proportional r_{prop} neighborhoods provide a more generic neighborhood selection process than hand tuning $k_{nearest}$ for each model. Once the local neighborhood is selected the distance between the current point and each neighbor is calculated. If the Cartesian distance is between r_{prop} and r_{min} , P_{new} is linearly interpolated and assigned the average of the two points' $\hat{\mathbf{n}}$ (Algorithm

6).

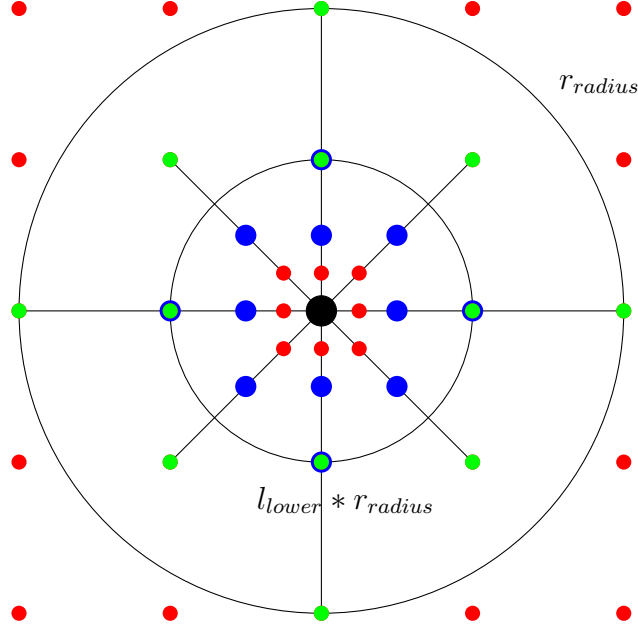


Figure 4.12: Visualization of a point cloud (red and green points), the current pose (black), the r_{radius} distance, the lower limit distance, the points included in interpolation (green), the points not included in interpolation (red), and the Algorithm 6 interpolated points (blue).

Concave regions, \mathbb{R}_{cave} , however, contain converging surface normals, $\forall P_{surf}.\hat{\mathbf{n}} \in \mathbb{R}_{cave}$, and require algorithmic strategies to manage increasing VF intralayer resolution. This can, if task parameters place a VF layer at the center, result in a VF layer with one Cartesian location but varying orientations (Figure 4.11, inner blue arrows). A hemisphere is a unique (but relevant) edge case, and interior corners and slots will require similar considerations. As with the task PCN, voxel filtering is one way to correct VF layer resolution. Voxel filtering discretizes space into boxes of a provided size and all points present within the same box are approximated with their centroid [Rusu, 2009]. For the TVF generation pipeline, the discretization box is a cube with side lengths proportional to d_{intra} .

Algorithm 6 Point Cloud Interpolation Calculation

```
1: warnings = 0
2: for  $P_{vf}(i); i = 1, \dots, I$  where I is the total points do
3:   Get neighbors within  $r = r_{prop}$  radius
4:   if  $0 < neighbors$  then
5:     for  $P_{vf}(j) : j = 1, \dots, J$  where J is the total neighbors do
6:        $d_{Cartesian} = \text{sqr}t((P_{vf}(i).v_x - P_{vf}(j).v_x)^2 + (P_{vf}(i).v_y - P_{vf}(j).v_y)^2 + (P_{vf}(i).v_z - P_{vf}(j).v_z)^2)$ 
7:       if  $d_{Cartesian} > r_{min}$  then
8:          $P_{new}.v = \begin{bmatrix} (P_{vf}(i).v_x + P_{vf}(j).v_x)/2, \\ (P_{vf}(i).v_y + P_{vf}(j).v_y)/2, \\ (P_{vf}(i).v_z + P_{vf}(j).v_z)/2 \end{bmatrix}$ 
9:          $P_{new}.\hat{n} = \begin{bmatrix} (P_{vf}(i).n_x + P_{vf}(j).n_x)/2, \\ (P_{vf}(i).n_y + P_{vf}(j).n_y)/2, \\ (P_{vf}(i).n_z + P_{vf}(j).n_z)/2 \end{bmatrix}$ 
10:        Add new point to point cloud
11:       end if
12:     end for
13:   end if
14: end for
```

VF layers are independently calculated $\forall P_{surf}(i).\hat{n}$ to remove the possibility of a chain of convergence or divergence distortions as layers are calculated at increasing distances from the task surface (Equation 4.5). Once constructed, the number of neighbors within a d_{intra} based r_{radius} volume of $P_{vf}(i)$ is calculated to check VF layer resolution. The number of neighbors is averaged over an entire VF layer to ensure the VF layer resolution is maintained as d_{layer} increases. Resolution checks added based on feedback from early TVF generation method evaluations [Sharp and Pryor, 2018].

It is important to note the goal is to maintain the direction and distance to fine surface features and create a PCN of extended surface features (Figure 4.11). This distinction mainly applies to interior features' extended points. At close distances to the surface the still looks

similar to the task surface (Figure 4.13, layers 1 and 2). However, as the distance increases the calculated PCN clusters in the center of the interior feature (Figure 4.13, layers 3). Eventually, the VF layer will approach the opposite side of the feature (Figure 4.14). In such cases, surface reconstruction methods would likely smooth away important information or result in non-manifold geometry [Botsch et al., 2010]. As such, interpolation and voxelization is executed only once per VF layer instead of iteratively to limit surface distortions in the resulting PCN. The effects of integrating spatially aware PF curvature estimation [Foorginejad and Khalili, 2014] into the interpolation process and the effects of reconstruction algorithms on non-manifold geometry based on task parameters are areas of future work.

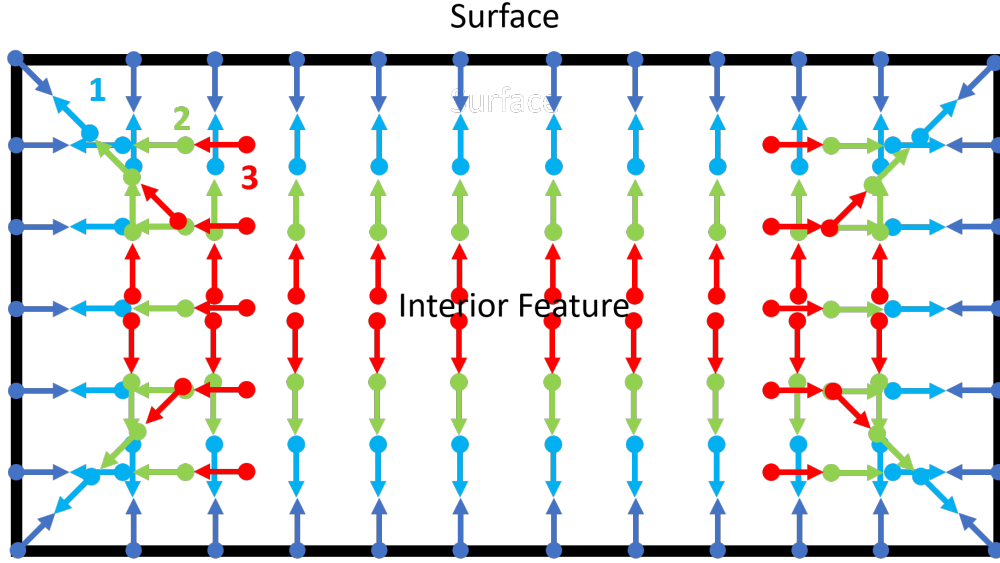


Figure 4.13: The first three VF layers in an interior slot.

The VF layer can now either become a GVF or FRVF as required by the task. For a GVF, normal vectors are determined by inverting each $P_{surf}(i) \cdot \hat{\mathbf{n}}$ to face toward the task

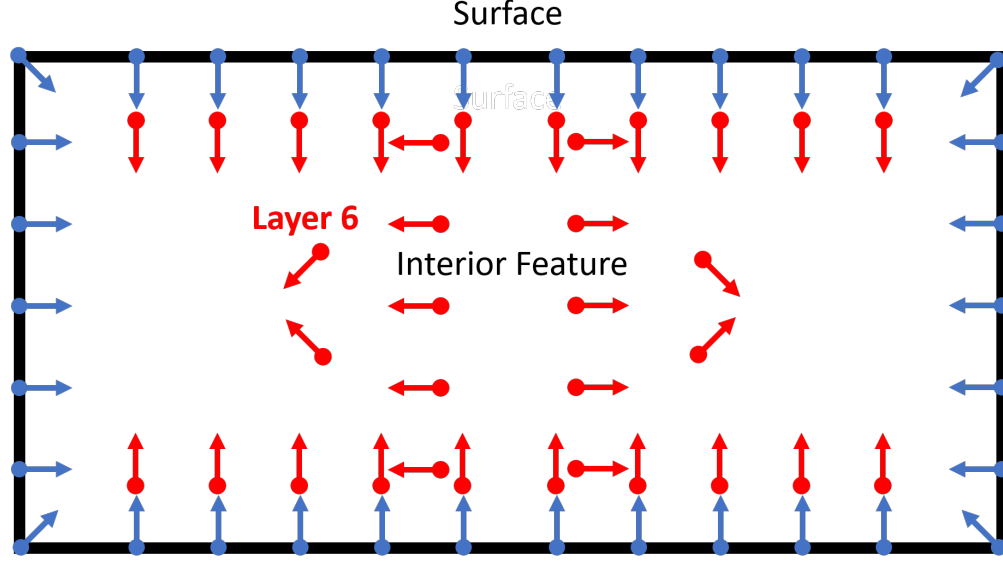


Figure 4.14: The sixth VF layer demonstrating how interior spaces can result in complex point clouds which are not conducive to conversion into a mesh.

surface (Equation 4.5). Conversion into a FRVF allows VF layers to become a protective FRVF outside the original task surface and may be desirable for small d_{min} non-contact tasks. FRVFs also handle cases where d_{max} is greater than an interior space by removing regions from operator use. FRVFs can be converted into a polygonal mesh (Figure 4.15) or each point can have a spherical forbidden region. As previously mentioned, VF layer generation can result in point clouds which are difficult to convert into polygonal meshes without generating non-manifold surfaces. It is therefore only recommended when the ratio of surface feature size to layer distance is high, such as with plasma cleaning.

The alternative to polygonal mesh conversion is to add a spherical forbidden region around each point (Figure 4.16). Spherical forbidden region point clouds were previously used in [Yamamoto et al., 2012; Kosari et al., 2014] for organ protection during surgery. This

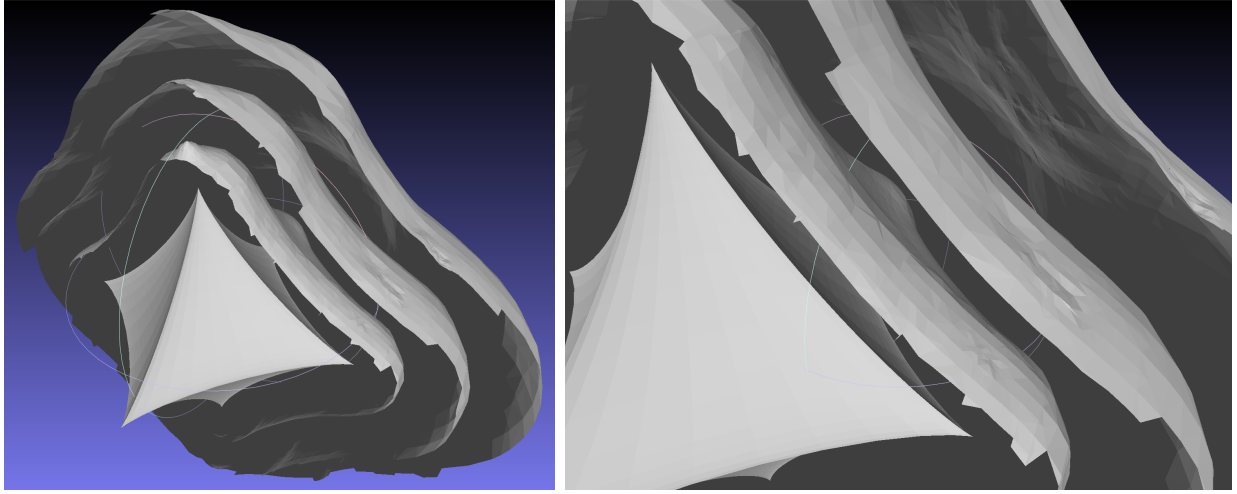


Figure 4.15: Three VF layers around a superellipsoid were converted into polygonal meshes to function as FRVFs. Half the FRVFs have been removed to visualize interior surfaces (left). A closeup of the superellipsoid and FRVF surfaces (right). This demonstrates conversion at multiple distances but only one FRVF is required for a task.

method is flexible but represents the forbidden region as many spherical meshes instead of a single mesh and will, therefore, have visualization limitations.

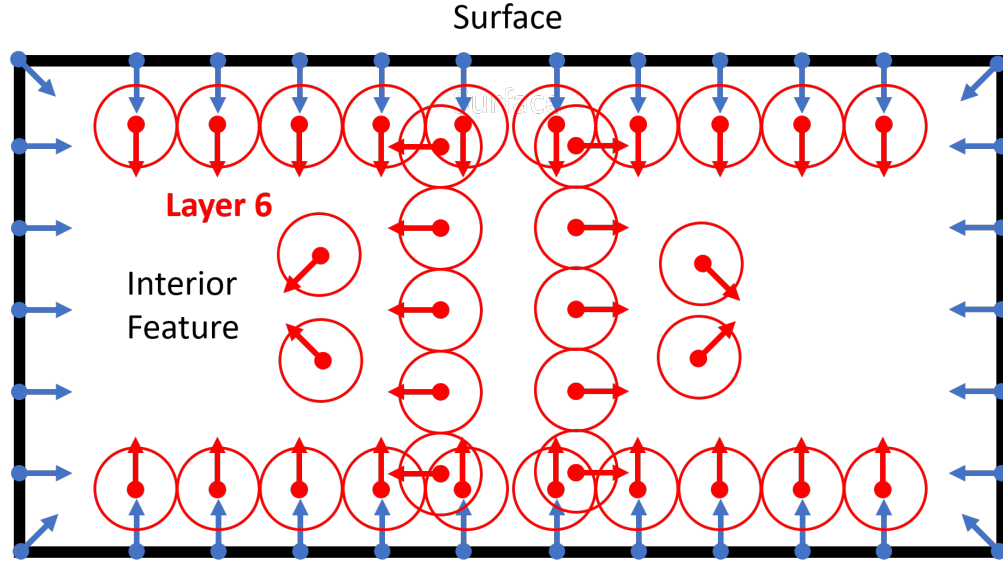


Figure 4.16: The sixth VF layer with spherical forbidden regions added.

4.5 Guidance Virtual Fixture Data Storage

During the extension of VF generation techniques, shortcomings in the previous VNSVF volumetric primitive approach were discovered [Sharp et al., 2018]. The first issue was array storage inefficiency due to the quantity of empty locations required to accommodate the largest VF dimensions. To address this shortcoming, GVF layer information is stored in a graph structure to allow future utilization of graph search tools such as Breadth First, Depth First, and Uniform Cost Search in addition to Dijkstra’s Shortest Paths and other path algorithms. A graph G consists of a set of vertices, V , which are connected in pairs and stored in a set of edges, E (Figure 4.17). Edges between vertices are bi-directional and stored in an adjacency list.

$$\begin{aligned}
G &= \{V, E\} \\
V &= \{A, B, C, D, E, F\} \\
E &= \{AB, AC, AD, AE, AF, BC, BD, CD\}
\end{aligned} \tag{4.6}$$

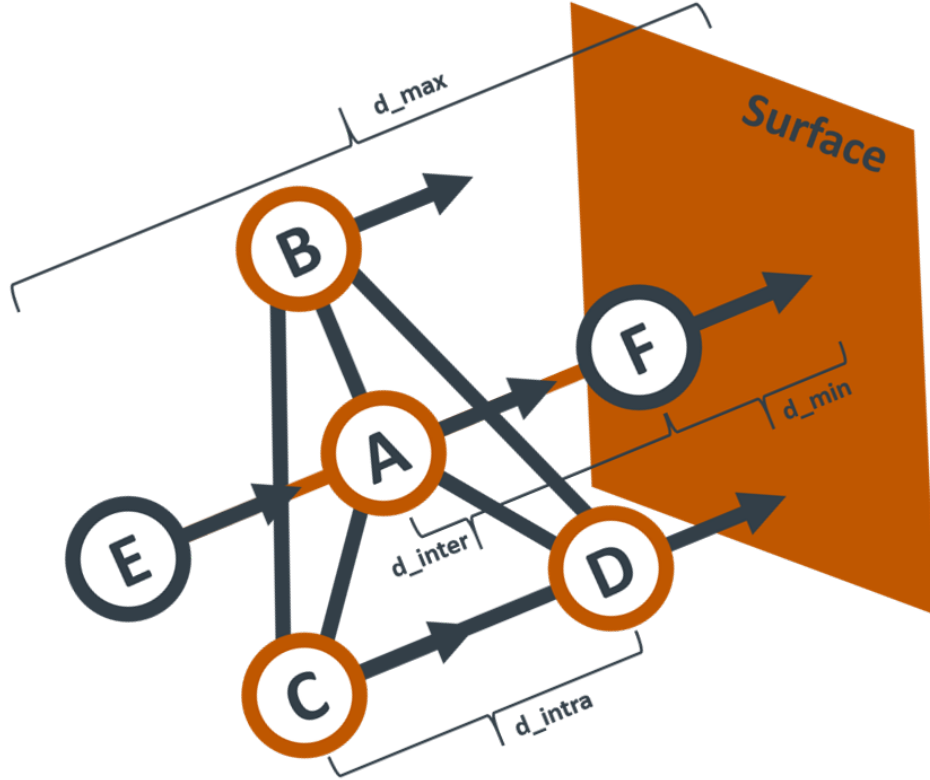


Figure 4.17: Bi-directional graph structure where the set of vertices is V and the set of edges is E (Equation 4.6). TVF task parameters are labeled and vertices A, B, C, D are in the same GVF layer, Sharp and Pryor [Sharp and Pryor, 2018].

VF surfaces are calculated based on minimum & maximum distances (d_{min} , d_{max}) and intralayer & interlayer distances (d_{intra} , d_{inter}) task parameters (Figure 4.17). The TVF graph is constructed by converting each GVF layer’s point cloud into vertices. Each vertex

contains GVF layer and 6 DOF pose in the TVF frame information but could include other data such as curvature and local spatial neighborhood information. Edge weights record Cartesian distance information but could also integrate weighting on multiple metrics such as Cartesian and angular distances. Complex or highly detailed models may require visualization limitations, perhaps based on distance from the current vertex, in order to reduce operator burden. TVFs can be created for multiple tasks on a single surface or *vice versa* and are stored for later use as the FRVF task surface model and the TVF graph.

4.6 Summary of Combination Guidance and Forbidden Region Virtual Fixtures

This chapter describes the task surface geometry approach to TVF construction based on a polygonal mesh FRVF and layers of point cloud GVFs. VF generation methods were advanced using, previously uninvestigated, point cloud GVFs to progress toward “a highly automated remote technology that can deliver a non contact smarter dismantling process, cut most materials, cut **complicated structural geometries...**” [Hilton and Khan, 2014]. Spatially discrete and spatially continuous tasks which are too complex for volumetric primitive approaches require algorithms applicable to data from multiple sources. Surface normals are calculated and extended to build layers of PCN VFs at varied offset distances from the task’s surface. These VF layers are then interpolated and voxelized to ensure intralayer task resolution is maintained. The VF layers can be converted into a FRVF or GVF layers. A bi-directional graph structure retains GVF information and is paired with the FRVF to form a TVF. These TVFs are more expressive than volumetric primitives thus addressing well-documented VF generation shortcomings.

Chapter 5

Task Virtual Fixture Construction Implementation

This chapter outlines the software implementation of TVF generation pipeline algorithms¹. The triangular polygonal mesh format which was chosen for pipeline input was the binary STL format (Listing 5.1), which is a widely available open source and commercial program output [MeshLab, 2018; PTC, 2017; SOLIDWORKS, 2017]. Generation pipeline software was written in C++ but includes several third party libraries such as Point Cloud Library (PCL), Boost Graph Library (BGL), and Open MP (OMP). TVF generation is independent of ROS with the exception of ROS information and error messages used during STL testing and TVF graph construction. A separate ROS, *RViz*, and *MoveIt!* interface was developed for visualization, control, evaluation. A conversion program was also created to provide TVF poses through a second visualization and control interface for ABB robots running proprietary software.

Code 5.1: STL File Format

```
solid name
facet normal nx ny nz
  outer loop
    vertex v1x v1y v1z
    vertex v2x v2y v2z
```

¹This chapter includes material published in Sharp and Pryor [2018] where I contributed that form of TVF generation pipeline, the object testing, and the analysis of the results.

```

        vertex v3x v3y v3z
    endloop
endfacet
endsolid name

```

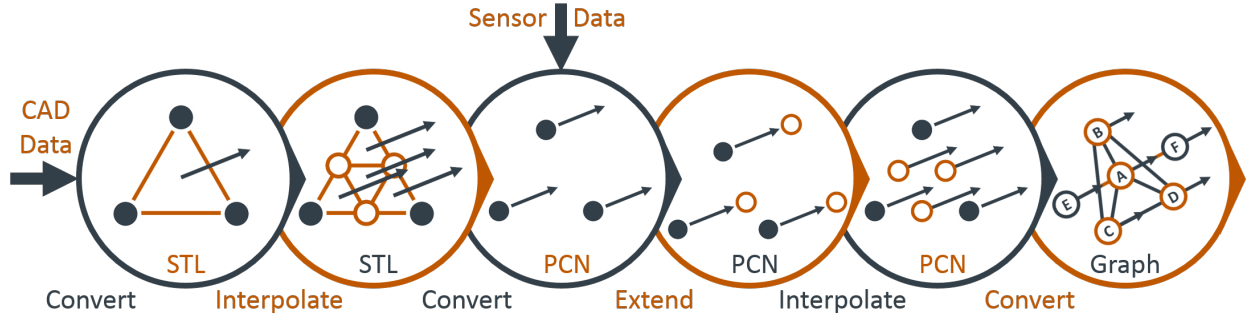


Figure 5.1: TVF generation algorithms to go from a polygonal mesh or sensor data to a bi-directional graph structure.

5.1 Task Surface Virtual Fixture Software Pipeline

Task surface information is read from binary STLs (Listing 5.1) into an array of triangles. The TVF generation pipeline mesh $\hat{\mathbf{n}}$ (Algorithm 3) threshold was set to $\varepsilon_1 = 1e-5$. Since $\hat{\mathbf{n}}$ are normalized this values is unit less. An operator warning is sent if meshes exceed ε_1 , suggesting average is skewed and the mesh is malformed. Once appropriate operator warnings are sent the array of triangles is interpolated based on task parameters (Algorithm 4). This interpolation process iterates through all triangles and therefore has $O(N)$ complexity.

After interpolation the triangle array is converted (Algorithm 5) into a Point Cloud with Normals (PCN). PF methods provided inconsistent results for models with varying point density and curvature (Figure 4.10, top and middle). To achieve more consistent results,

points are assigned the polygonal mesh calculated surface normal (Figure 4.10, bottom). During PCN conversion, point $\hat{\mathbf{n}}$ s are checked for malformed meshes (Algorithm 5). The ε_2 threshold was set to unit less value $1e - 8$. At this threshold, a bowl model [Men in Black, 2018], known to have duplicate inverted triangles, warned the operator 100% of points were below the ε_2 threshold.

The PCN conversion process creates a PCN point for each triangle vertex resulting in duplicate points equal to the number of triangles in which a vertex is present. To removed duplicate points the PCN is voxelized with a leaf size equal to one tenth of the task d_{intra} to scale with task parameters instead of using a static value. Once converted to a PCN and voxelized the data is saved as a PCD file (Listing 5.2).

Code 5.2: PCD File Format

```
# .PCD v.7 - Point Cloud Data file format
VERSION .7
FIELDS x y z rgb
SIZE 4 4 4 4
TYPE F F F F
COUNT 1 1 1 1
WIDTH 213
HEIGHT 1
VIEWPOINT 0 0 0 1 0 0 0
POINTS 213
DATA ascii
0.93773 0.33763 0 4.2108e+06
0.90805 0.35641 0 4.2108e+06
.
.
.
```

The PCN is created, voxelized, and saved using Point Cloud Library (PCL). PCL

is an open source library for point cloud processing, “The library contains state-of the art algorithms for: filtering, feature estimation, surface reconstruction, registration, model fitting and segmentation. PCL is supported by an international community of robotics and perception researchers” [Rusu and Cousins, 2011]. Additional PCL capabilities include point cloud visualization and estimating surface normals for point clouds (Figure 5.2). PCL integration into the TVF pipeline allows sensor data utilization after the PCN conversion step (Figure 4.1).

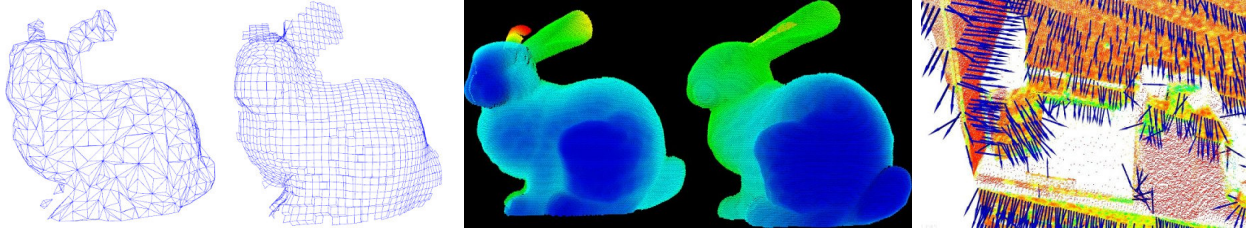


Figure 5.2: Surface meshing (left) [Rusu and Cousins, 2017b], Visualization (middle) [Rusu and Cousins, 2017b], Surface normal estimation viewpoint corrected (right) [Rusu and Cousins, 2017a]

Once converted into a PCN, point normals are extended based on task parameters to create VF layers (Equation 4.5). These layers are then interpolated and voxelized, based on the task r_{prop} distance, to maintain surface density (Algorithm 6). Due to the lack of spatial information in an unorganized PCN, the interpolation algorithm requires nested ‘for’ loops and therefore has $O(N^2)$ complexity.

The final TVF generation step is to convert GVF PCN layers into a graph structure. Since vertices represent spatial locations instead of directional paths a bi-directional graph was chosen to maintain greatest future flexibility. Graph vertices represent 6 DOF EEF poses in the task frame and are indexed by an ID. Edge weights record Cartesian distance

information but could also integrate weighting of multiple metrics such as Cartesian and angular distances.

Vertices are graphically linked in two different ways. Each point in GVF layer is connected to all other points in the GVF layer. In other words PCN layers are fully connected (Figure 4.17, Edges AB , AC , AD , BC , BD , CD). The number of edges is therefore dependent on the number of vertices and grows at a known rate (Equation 5.1).

$$E_{intra\text{layer}} = (N_{\text{layer vertices}}^2 - N_{\text{layer vertices}})/2 \quad (5.1)$$

Connections between GVF layers are more sparse (Figure 4.17, Edges AE , AF). GVF layer points are only connected to the closest point in the neighboring GVF layers (Algorithm 7). Consider a TVF graph containing three GVF layers. This approach means vertices in layer one will connect to vertices in layer two only, since there is no GVF layer closer to the task FRVF surface. Vertices in layer two, however, will connect to vertices in both the next inner and outer layers from the current task FRVF surface. The third layer is connected similarly to layer one but without a layer farther from the task FRVF surface is connected only to a layer closer to the surface. This algorithm also has $O(N^2)$ complexity due to the need to iterate through both PCN layers during linking (Algorithm 7).

Linking between GVF layers can result in more than one inter-layer edge per PCN point. For example, in Figure 4.11, outer layers, converging surface normals would cause multiple vertices in the blue layer to link to the same vertex in the green layer or *vice versa* respectively. This interlayer linking method reduces overall TVF graph size and simplifies operator navigation. Combination of GVF PCN layers into a bi-directional graph was written

Algorithm 7 Graph Interlayer Linking

```
1: for  $P_{vf}(i); i = 1, \dots, I$  where I is the total points in a GVF layer do
2:   Clear  $d_{min}, index_{min}$ 
3:   for  $P_{vf}(j); j = 1, \dots, J$  where J is the total points in a neighboring GVF layer do
4:      $d_{Cartesian} = \text{sqrt}((P_{vf}(i).v_x - P_{vf}(j).v_x)^2 + (P_{vf}(i).v_y - P_{vf}(j).v_y)^2 + (P_{vf}(i).v_z - P_{vf}(j).v_z)^2)$ 
5:     if  $d_{min} > d_{Cartesian}$  then
6:        $d_{min} = d_{Cartesian}$ 
7:        $index_{min} = j$ 
8:     end if
9:   end for
10: end for
11: if No edge between  $i$  and  $index_{min}$  then
12:   Add interlayer Edge to graph
13: end if
```

using the C++ Boost Graph Library (BGL) [Boost, 2017]. BGL integration allows future utilization of graph search tools such as Breadth First, Depth First, and Uniform Cost Search in addition to Dijkstra’s Shortest Paths and other path algorithms.

5.1.1 Code Optimization

Algorithms at each stage of the software pipeline were multi-threaded using OpenMP [OpenMP, 2018] in order to reduce TVF generation time. These algorithms include polygonal mesh checking (Algorithm 3), triangle interpolation (Algorithm 4), polygonal mesh to PCN conversion (Algorithm 3), point normal extension (Equation 4.5), VF layer interpolation and voxelization (Algorithm 6), and graph interlayer linking (Algorithm 7). In each of these processes, calculations can be completed independently but with protected data recording which makes multi-threading effective at increasing TVF generation operational speed. TVF pipelines stages were also designed to be operated independently. This allows for further

analysis of task parameter variation during STL testing, GVF resolution examination, or output TVF graph size.

5.2 Task Virtual Fixture Visualization

Once the TVF generation software pipeline was complete, the output TVF graph must be visualized for operator utilization and evaluation. As with VNSVF development, TVFs are applicable to a variety of spatially discrete and spatially continuous tasks. Spatially continuous tasks include radiation surveys and visual inspection while spatially continuous tasks include painting, plasma cleaning, and laser cutting. Therefore, two methods were developed to fulfill visualization and control research requirements. The first visualization method integrated the TVF graph structure into a ROS and *RViz* visualization. A second visualization method allowed TVF poses to be loaded into ABB’s proprietary software, RobotStudio [ABB, 2018b].

5.2.1 Robot Operating System Visualization and Control Interface

To develop the TVF generation software pipeline the output TVF graph needed to be visualized. Therefore, a ROS and *RViz* interface was developed to display TVF information similarly to the previous VNSVF approach. This interface displays the task FRVF surface and allows TVF graph exploration. The operator’s current TVF vertex is maintained and neighboring vertices, including those in other layers, are displayed.

During the development of the TVF generation software pipeline, shortcomings in the previous VNSVF navigation methods were noted [Sharp et al., 2018]. The previous interface used another window to provide augmented input ‘Left’, ‘Right’, ‘Up’, ‘Down’,

‘In’, and ‘Out’ buttons for VNSVF navigation (Figure 3.12). One issue was the increased ambiguity of the augmented input buttons to operators with complex models [Sharp and Pryor, 2018]. With a spherical model, for example, operator movement up and over the top will provide a different viewpoint than moving around the side. Redesigning the directional input buttons and viewpoint following interface alleviated these issues. Design of the new control interface followed guidelines from analysis of eight of the 15 DARPA Robotics Challenge Trial teams including ‘more sensor fusion, fewer operators, and more automation lead to better performance’ [Yanco et al., 2015]. Removing the additional window containing the directional buttons in favor of interactive markers in the main *RViz* window improves sensor fusion. Each TVF pose is an interactive marker and right-clicking on it brings up a menu of control options. During operator evaluation of this interface, only one operator is expected at a time. Underlying ROS packages provide additional automation capabilities.

Instead of using the augmented input ‘Left’, ‘Right’, ‘Up’, ‘Down’, ‘In’, and ‘Out’ buttons the operator right-clicks on the desired marker and selects **Make current pose** (Table 5.1). Much like the VNSVF approach, the current TVF vertex is displayed as a white marker and moves with operator input (Figure 5.3). Interlayer linking is displayed with blue markers and changing the current pose to a blue marker changes the GVF layer (Figure 5.3). All TVF markers have a fixed Cartesian location in the task frame but can be rotated around their surface normal (Figure 5.3, left vs. right). The orientation change can be recorded in the TVF graph structure by clicking **Update pose** in the menu (Table 5.1).

During visualization environment construction, several options for TVF layer visualization were developed since visualizing the entire TVF graph would likely be incomprehensible. The first displayed all vertices neighboring the current TVF vertex (Figure 5.4, left). This

Table 5.1: Manipulator to Task Transform Tool Menu Options

Make current pose:	Selected marker becomes the current vertex and update the nearest neighbor graph and interlayer linking markers.
Update pose:	Update the TVF graph with the selected interactive marker rotation.
Move to pose:	Plan and execute EEF motion to the selected interactive marker pose.
Add pose to path:	Add the selected vertex to the graph of nodes for path motion.
Remove pose from path:	Remove the selected vertex from the graph of nodes for path motion.
Test path:	Pass the path graph to <i>Descartes</i> planning and execution package [Ratnesh Madaan, 2015].
Clear path:	Clear the graph of nodes for path motion.

option was effective for small TVF layers but became increasingly confusing as TVF layer size increased. For the second and third options, all interlayer linked vertices were displayed to the operator to preserve the ability to change TVF layer and displayed vertices are written to a separate graph structure to preserve future possibilities, such as automated path planning. The second and third display options vary in the selection of the intralayer linked vertices to display. Option two selects an operator defined $k_{nearest}$ neighbors, based on graph edge weights, from the current TVF vertex (Figure 5.4, middle). Displaying $k_{nearest}$ neighbors limits the confusion caused by increasing TVF layer size. Although, navigation across an area much larger than the $k_{nearest}$ neighbor size becomes increasingly tedious with increasing TVF layer size. Thus, a third option was developed where vertices are less likely to be displayed with increasing edge weight to the current vertex. All vertices closer than the task d_{intra} are displayed. However, at distances beyond this threshold, the likelihood decreases with edge

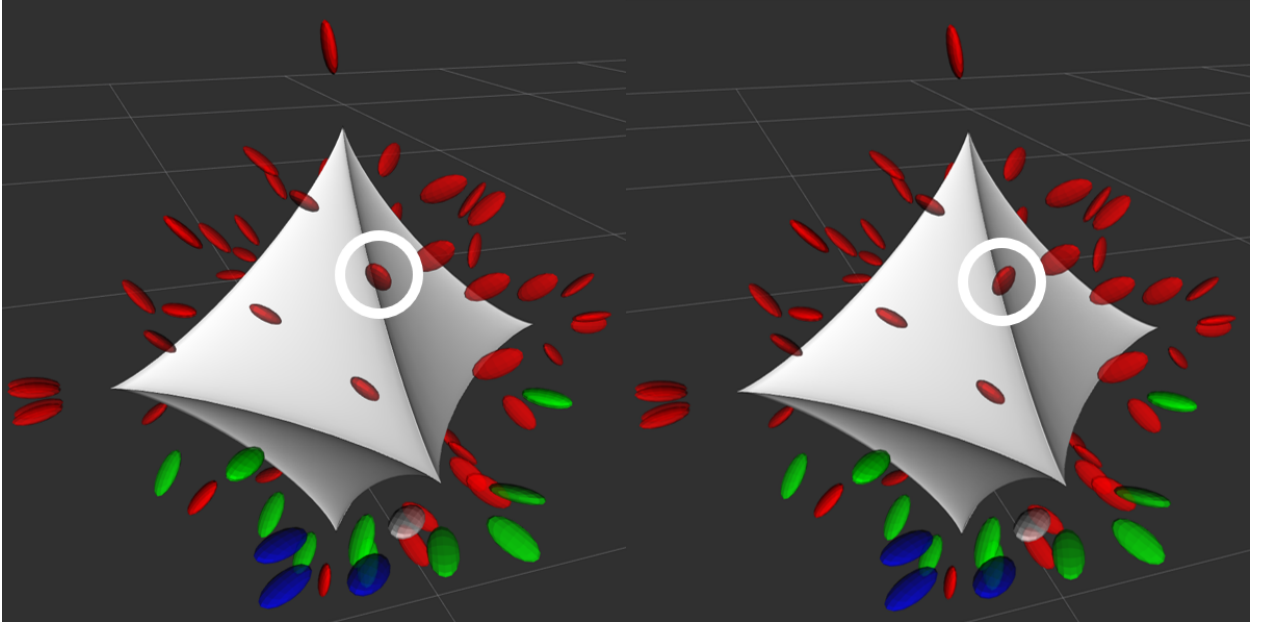


Figure 5.3: TVF visualization and control interface showing current vertex as a white marker and blue markers represent different GVF layers. The marker in the white circle has rotation about its normal altered by the operator between the left and right images.

weight (Figure 5.4, right). For each vertex sharing the current vertex's layer, the ratio of the task d_{intra} and edge weight between the vertices, $E_{i,j}$, is multiplied by a randomly generated number between one and 100. If the resulting value is above 25, the neighboring vertex is added to the display graph (Algorithm 8). Visualization algorithm option three results in a 25 % chance of display just above the d_{intra} threshold which then decreases linearly with edge weight.

Algorithm 8 Edge Weight Decreasing Graph Visualization

```
1: Current TVF pose =  $P_{tvf}(i)$ 
2: for  $P_{tvf}(j); i = 1, \dots, J$  where  $J$  is the current vertex's neighbors do
3:   if  $P_{tvf}(i).layer \neq P_{tvf}(j).layer$  then
4:     Add  $P_{tvf}(j)$  to display graph
5:   else if  $E_{i,j} > d_{intra}$  then
6:     Add  $P_{tvf}(j)$  to display graph
7:   else if  $rand * \frac{d_{intra}}{E_{i,j}} > 25$  then
8:     Add  $P_{tvf}(j)$  to display graph
9:   end if
10: end for
11: Display graph for operator
```

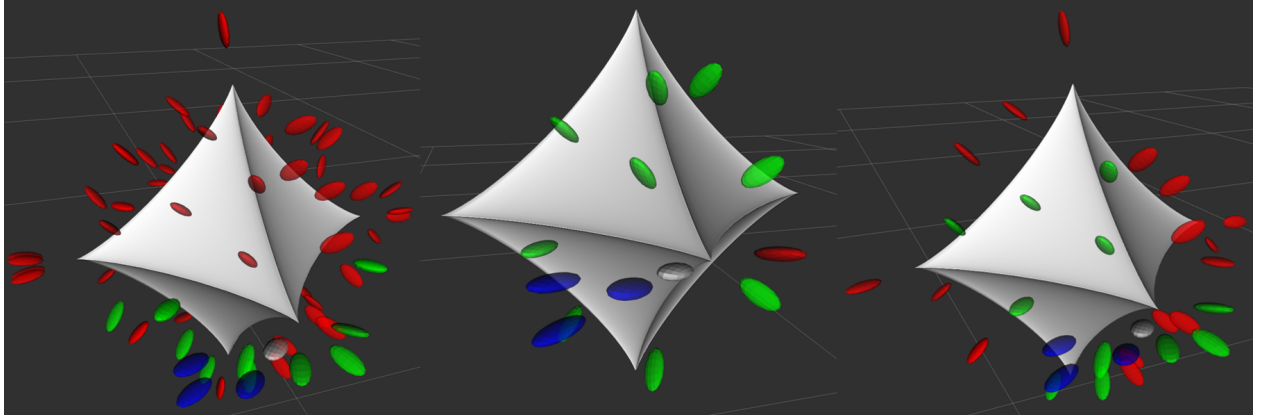


Figure 5.4: Display of the three developed TVF graph visualization methods: full GVF layer visualization (left), $k_{nearest} = 10$ neighbors (middle), edge weight decreasing (right). Reachability varies due to changes in task surface location.

5.2.1.1 Task Virtual Fixture Discrete Control Interface

As with VNSVF development, the new control and visualization interface was originally considered for assistance with spatially discrete non-contact tasks such as visual inspection. Therefore, the first required control was the ability to move to desired TVF poses. Users can

attempt to move the robot EEF to a specific TVF vertex by right-clicking on the interactive marker, as with **Make current pose**, and selecting **Move to pose**, (Table 5.1). Motion to the pose will be successful if the pose has an IK solution and the robot is unobstructed by the planning scene.

The tedium of manually testing poses led to the integration of reachability analysis into the TVF visualization interface. The reachability analysis includes an IK check, through Kinematics and Dynamics Library (KDL) [The Orocos Project, 2015a], and a planning scene check, with Open Motion Planning Library (OMPL) [Sucan et al., 2012], to verify a collision-free solution. Both KDL and OMPL are part of the underlying automation capabilities of *MoveIt!*. TVF vertices are green (reachable) or red (not reachable) in the operator interface ((Figure 5.5, left).

The task surface representation was expanded beyond the VNSVF approach to provide additional task frame freedom during workspace reachability analysis. The TVF visualization interface includes the task surface as an FRVF and an interactive marker. This means the operator can click and drag the task surface to the desired location in the virtual manipulator’s workspace (Figure 5.5, right). The interface also aggregates reachability analysis information and displays the number and percentage of TVF vertices along with the current task frame pose (Figure 5.6, white writing). Each control interface can be activated or removed using controls in the *RViz* window (Figure 5.6, left side).

Flexible positioning and reachability testing interface expansions create a tool allowing the operator to test many locations and determine an acceptable transform from the manipulator base to the task surface to achieve task completion. As such, the tool was named the Manipulator to Task Transform Tool (MTTT). The overall process is similar to determining

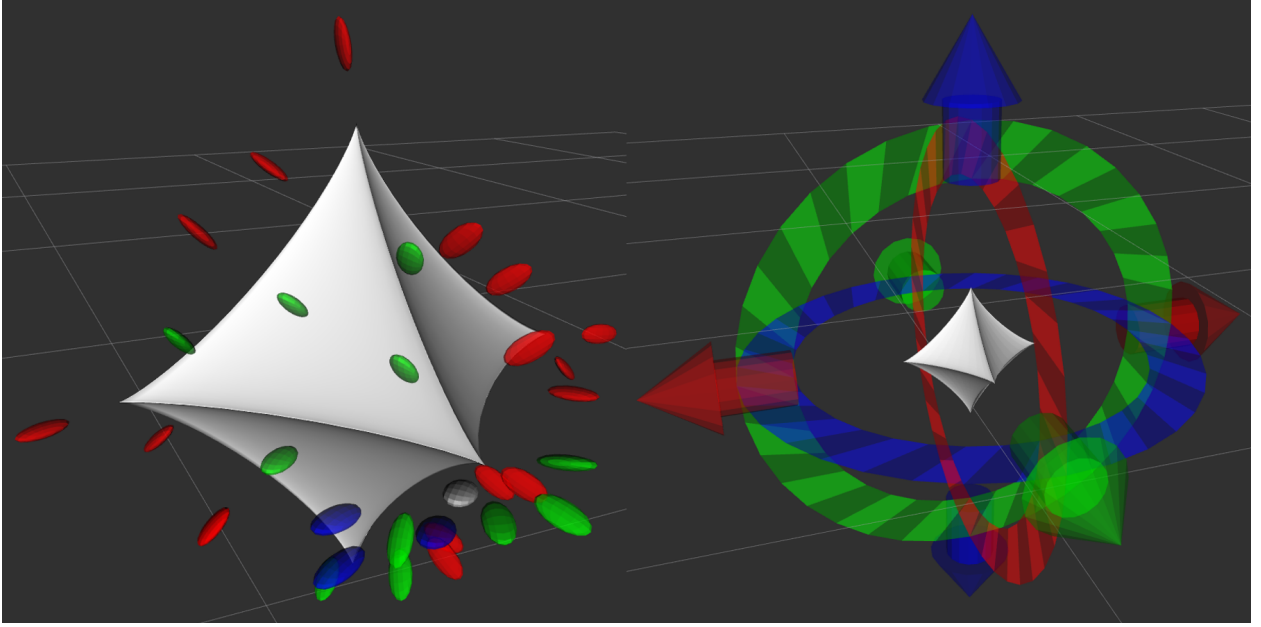


Figure 5.5: TVF vertex reachability analysis visualization with the current vertex (white), different GVF layer vertices (blue), same GVF layer reachable vertices (green), and same GVF layer unreachable vertices (red) (left). Visualization of the task FRVF surface and 6DOF interactive marker control (right).

mobile manipulator base placement for task completion using volumetric primitives presented in Section 3.2.1. However, it is much more functional since operators can continuously place and test new locations, instead of batch testing of discrete grid placement. The updated display is also updated continuously and more generic. The tool is also independent of TVF generation. Thus, it provides task placement benefits for data from any source after conversion to an appropriately structured graph.

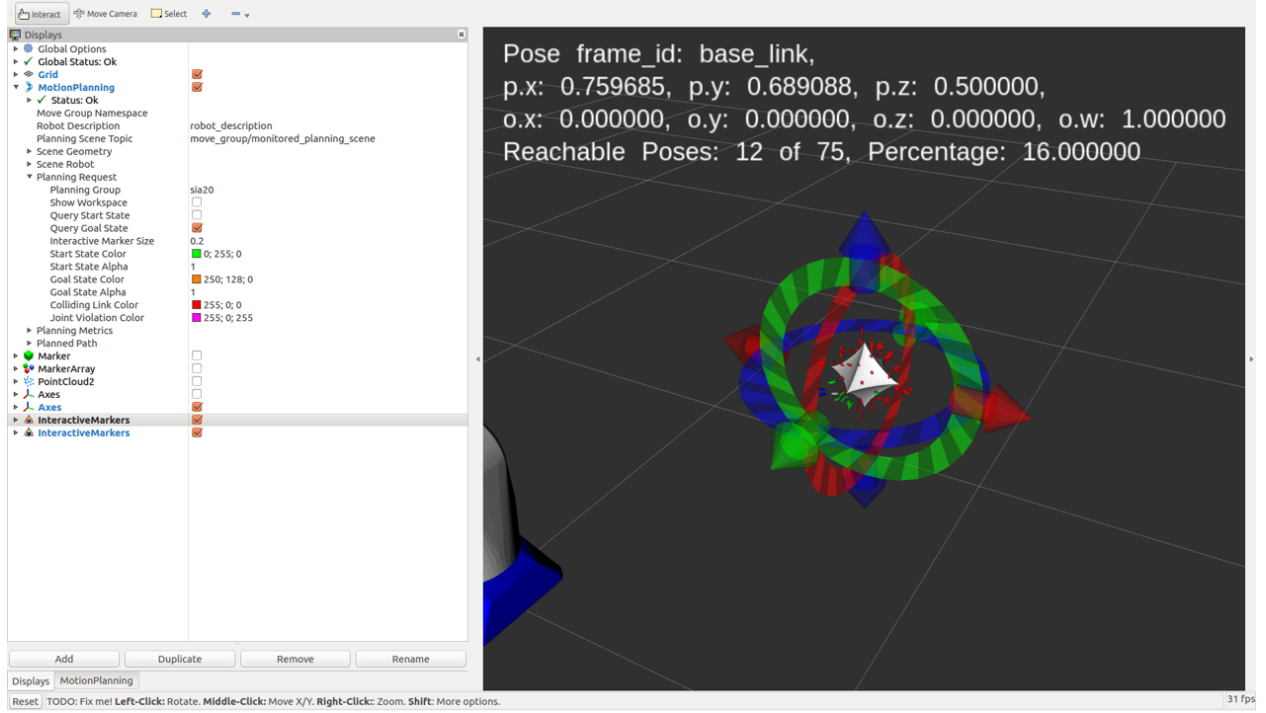


Figure 5.6: Full spatially discrete task *RViz* visualization and control interface showing the task FRVF, task surface interactive marker, TVF vertex reachability analysis, and *RViz* visualization controls.

5.2.1.2 Task Virtual Fixture Continuous Control Interface

In addition to integrating spatially discrete task control, spatially continuous task control was also integrated into the MTTT. As with VNSVFs, such tasks include plasma cleaning and laser cutting. Path planning through the ROS package *Descartes* [Edwards, 2015a] was continued from the second iteration of the VNSVF control interface [Sharp et al., 2018].

The high-level interface changed very little during the addition of spatially continuous task control (Figure 5.6). TVF markers allow the addition, removal, testing, and clearing

of a task path through the appropriate menu option (Table 5.1). Continuous task path information is displayed for the operator through teal colored TVF markers and directional arrows with the shade lightening as the path proceeds (Figure 5.7 bottom middle). Much like displayed vertex information, path information is stored in another TVF graph structure for future utilization of Dijkstra’s Shortest Path and other algorithms to improve execution efficiency. Task paths can be tested similarly to poses with *Descartes*. TVF poses in the path plan failing a *Descartes* check are colored black for the operator (Figure 5.7 bottom right). Failures result from several possible reasons including poses being outside the manipulator’s workspace or collisions with environmental obstacle. Failure causes are usually apparent from reachability analysis of the path poses.

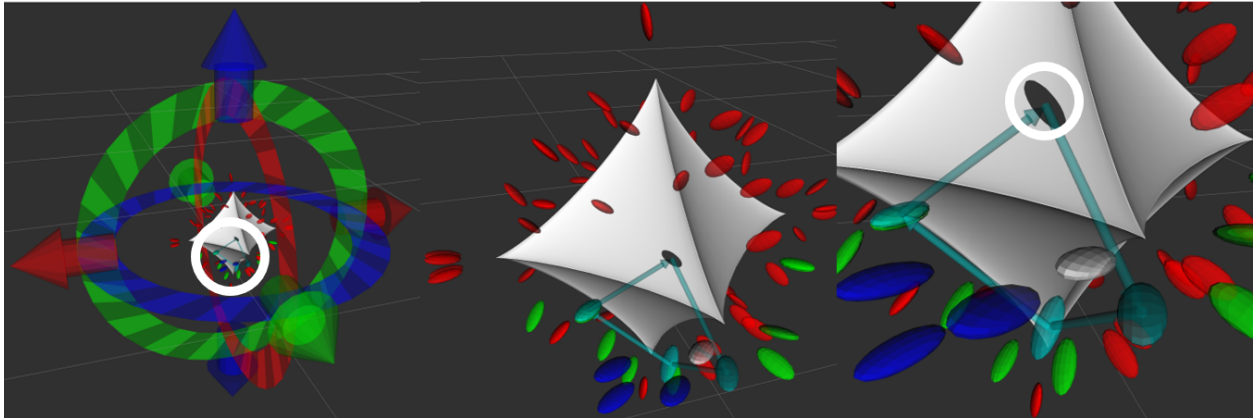


Figure 5.7: Close up visualizations of the displayed path graph (left, middle) and highlighted failure vertex (right, white circle).

Task path testing for feasibility using the *Descartes* package does have some limitations. One of the most significant is joint reconfigurations between two TVF poses in the task path. During this process, the IK solution and planning scene check at each of the TVF poses

may be valid. However, the EEF path during the reconfiguration collides with the robot or environment [ROS Industrial Consortium, 2018]. While this problem is uncommon, it is a possibility. Therefore, *Descartes* was acceptable for spatially continuous tasks as long as paths were verified before execution but unacceptable for user trials with industrial manipulator hardware.

5.2.2 ABB’s RobotStudio Visualization and Control Interface

The uncertainty during *Descartes* joint reconfigurations and following national laboratory safety concerns led to the integration of a second visualization and control interface. This second interface allows operators to utilize and evaluate TVF poses in ABB’s proprietary RobotStudio [ABB, 2018b] and execute industrially robust paths on ABB hardware (Figure 5.8).

Unlike opensource ROS and *RViz* software, bi-directional graph integration into the proprietary RobotStudio visualization environment is infeasible. Therefore, GVF PCN layer information is written to an ABB format prior to conversion into a TVF. Each GVF pose is converted into a ‘robtargt’ which is how ABB stores EEF pose and robot configuration information [ABB, 2018c]. GVF pose information is separated into Cartesian coordinates and quaternion information, which fills the first and second arrays of a robtarget’s four array format (Listing 5.3). The third and fourth array sections contain robot configuration and external axis information. The robot configuration array defines the manipulator’s axis configuration and is dependent on both the hardware and the chain of transforms from the robot base to the pose. External axis information describes the rotary or linear position of additional hardware controlled by the same controller as the manipulator. Examples of such

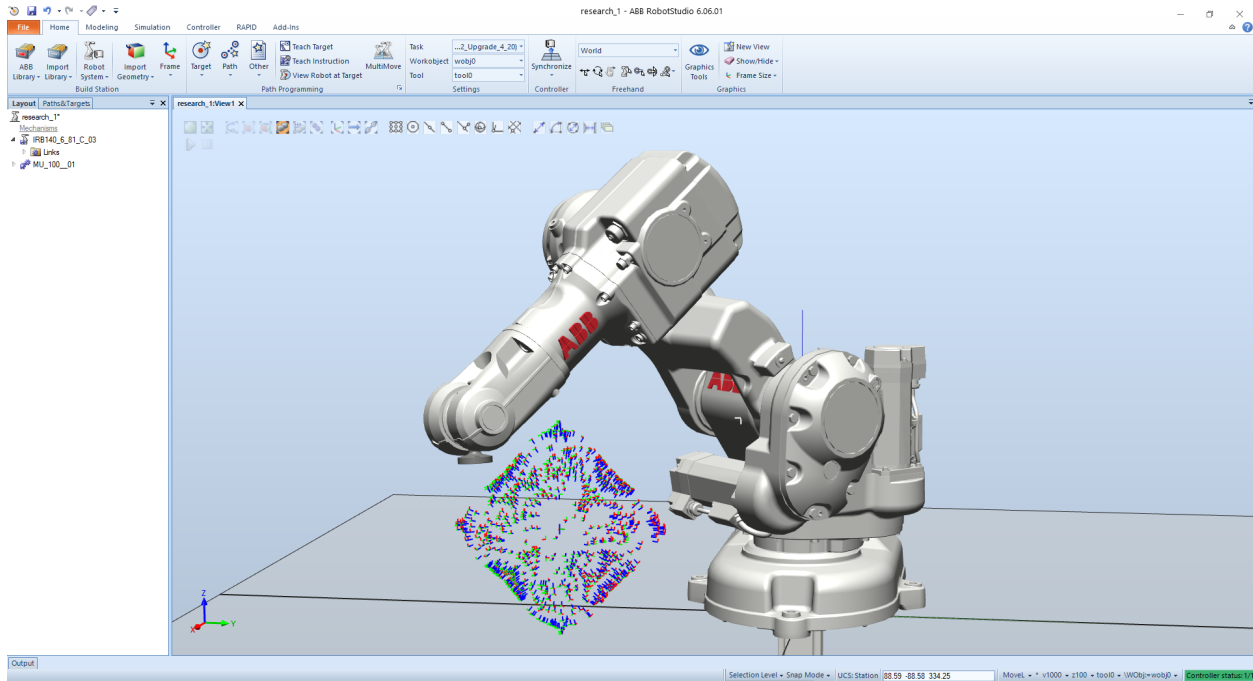


Figure 5.8: Example of TVF poses loaded into ABB’s RobotStudio [ABB, 2018b] for operator utilization.

hardware include EEF tools and rotary or linear drive systems. Both robot configuration and external axis information are hardware and environmentally dependent.

Code 5.3: ABB robtargt format.

```
< trans of pos >
  < x of num >
  < y of num >
  < z of num >
< rot of orient >
  < q1 of num >
  < q2 of num >
  < q3 of num >
  < q4 of num >
```

```

< robconf of confdata >
  < cf1 of num >
  < cf4 of num >
  < cf6 of num >
  < cfx of num >
< extax of extjoint >
  < eax_a of num >
  < eax_b of num >
  < eax_c of num >
  < eax_d of num >
  < eax_e of num >
  < eax_f of num >

```

GVF layers information is transferred into individual files of robtargets instead of a single TVF graph to visualize GVF layers independently. The ABB file type is a MOD file and contains a program module, ‘MODULE m’, which loads into RobotStudio and onto ABB hardware (Listing 5.4, first and final lines). Individual robtargets are defined as constants to avoid pose destruction during task program creation. Robtargets are named with the same ID, ‘pTVFID’, assigned during GVF layer to TVF graph conversion to avoid confusion. Robot configuration and external axis values remain as default for adjustment based on the hardware and environment where the TVF information will be used (Listing 5.4).

Code 5.4: MOD file format for ABB testing.

```

MODULE mTVF(SYSMODULE)
  CONST robtarget pTVF0:=[
    [-60.00,-40.00,-4.00], -> x,y,z
    [-0.251,0.067,0.251,0.932], -> q1,q2,q3,q4
    [0,0,0,0], -> robot configuration
    [9E+09,9E+09,9E+09,9E+09,9E+09,9E+09] -> external axes
  ];
  CONST robtarget pTVF1:=[
    [-58.00,-39.00,-3.50], -> x,y,z

```

```

[ -0.251,0.067,0.251,0.932] , -> q1,q2,q3,q4
[0,0,0,0] , -> robot configuration
[9E+09,9E+09,9E+09,9E+09,9E+09,9E+09] -> external axes
];
.
.
.
ENDMODULE

```

One of the advantages of the Manipulator to Task Transform Tool is the ability to move the task surface based on operator preference or environmental information. This capability is maintained in the ABB visualization and control interface but transforms must be adjusted prior to testing much like the VNSVF approach. ABB motion commands (Listing 5.5, MoveJ) allow the pose frame to be defined. This goal frame is referred to as a work object (Listing 5.5, 'WObj'). Thus, manipulator motions using TVF poses must have the work object defined as the task frame (Listing 5.5, WObj:=wobjTVF). The transform from the world frame to the task surface work object is defined for task execution by environmental information much like robot configuration.

Code 5.5: ABB joint move example where the work object is defined.

```
MoveJ pTVF0, v1000, z0, PlasmaPen\WObj:=wobjTVF;
```

5.3 Task Virtual Fixture Implementation Conclusions

This chapter outlines the software implementation of the TVF generation pipeline and visualization tools. The triangular polygonal mesh format chosen for pipeline input was the binary STL format. TVF software was written in C++ with the use of third-party libraries including Point Cloud Library (PCL), Boost Graph Library (BGL), Open MP (OMP), ROS,

RViz, and *MoveIt!*. Fully connected intralayer and sparse interlayer graph vertex linking were detailed. Efforts to optimize and multithread TVF generation algorithms were also outlined.

Two visualization environments with differing advantages were developed. A ROS, *RViz*, and *MoveIt!* tool provides full TVF graph visualization, TVF pose reachability testing, spatially discrete task execution, and path checking. This environment was named the Manipulator to Task Transform Tool due to its ability to aid operators in task placement in the manipulator workspace for task completion. Due to joint reconfiguration uncertainty during path execution, *Descartes* was deemed unacceptable for hardware execution of task paths. Thus, a second visualization and control environment was developed based on ABB control software. ABB's proprietary RobotStudio software limited reachability testing and interactive task surface location testing but provided industrially robust hardware execution of task paths. TVF generation software pipeline and control environment implementations provide a basis for algorithm testing and operator evaluation which is described in the following chapter.

Chapter 6

Task Virtual Fixture Evaluation

This chapter presents TVF generation analysis and operator evaluation. First, a variety of task surface models provide statistical information to verify VF layer resolution effects. Next, operators test TVFs on a spatially discrete task using the MTTT. The final evaluation has operators perform a spatially continuous task on ABB hardware. These tests intend to demonstrate TVF applicability to a wide variety of tasks.

6.1 Task Virtual Fixture Evaluation for General Surfaces

The first stage of TVF generation evaluation is to test on surface models drawn from multiple sources including mathematically constructed parametric surfaces, crowdsourced meshes from an online 3D model repository, and LIDAR PCN data. For each input data type the polygonal meshes are tested (if applicable), VF layers generated, VF layer resolution checked, and GVF layers converted into a TVF graph. The goal of this analysis is to verify VF layer resolution effects at increasing distances from the task surface and TVF graph vertices/edges grow at expected rates.

6.1.1 Parametric Surface Task Virtual Fixture Generation

Superellipsoids (Equation 6.1) and supertoroids (Equation 6.2) are mathematically generated parametric surfaces with multiple continuous parameters, $N_{xy}, N_z \in [0, \infty)$, producing a multi-manifold continuum of 3D objects (Figures 6.1, 6.2). Thus, they provide a generalizable, repeatable data set for testing and validation inclusive of edge cases. The constant parameters are set equal to one, $r_x, r_y, r_z, r_0, r_1 = 1$, resulting in models approximately 2 m across. Generation of the superellipsoids and supertoroids took place in the parameter space $0 \leq N_z, N_{xy} \leq 4$ at intervals of 2 providing nine of each for TVF generation pipeline evaluation. Model STLs (Figures 6.1, 6.2) were created with the Visualization Toolkit (VTK). VTK is a freely available software system for 3D computer graphics, image processing, and visualization [The Visualization Toolkit, 2018].

$$\begin{aligned}
 x &= r_x * \cos \theta^{N_z} * \cos \beta^{N_{xy}} \\
 y &= r_y * \cos \theta^{N_z} * \sin \beta^{N_{xy}} \\
 z &= r_z * \sin \theta^{N_z}
 \end{aligned} \tag{6.1}$$

$$\frac{-\pi}{2} \leq \theta \leq \frac{\pi}{2}, -\pi \leq \beta \leq \pi, 0 \leq N_{xy}, N_z < \infty$$

$$\begin{aligned}
 x &= \cos \theta^{N_{xy}} * (r_0 + r_1 * \cos \phi^{N_z}) \\
 y &= \sin \theta^{N_{xy}} * (r_0 + r_1 * \cos \phi^{N_z}) \\
 z &= \sin \phi^{N_z}
 \end{aligned} \tag{6.2}$$

$$0 \leq \theta \leq 2 * \pi, 0 \leq \phi \leq 2 * \pi, 0 \leq N_{xy}, N_z < \infty$$

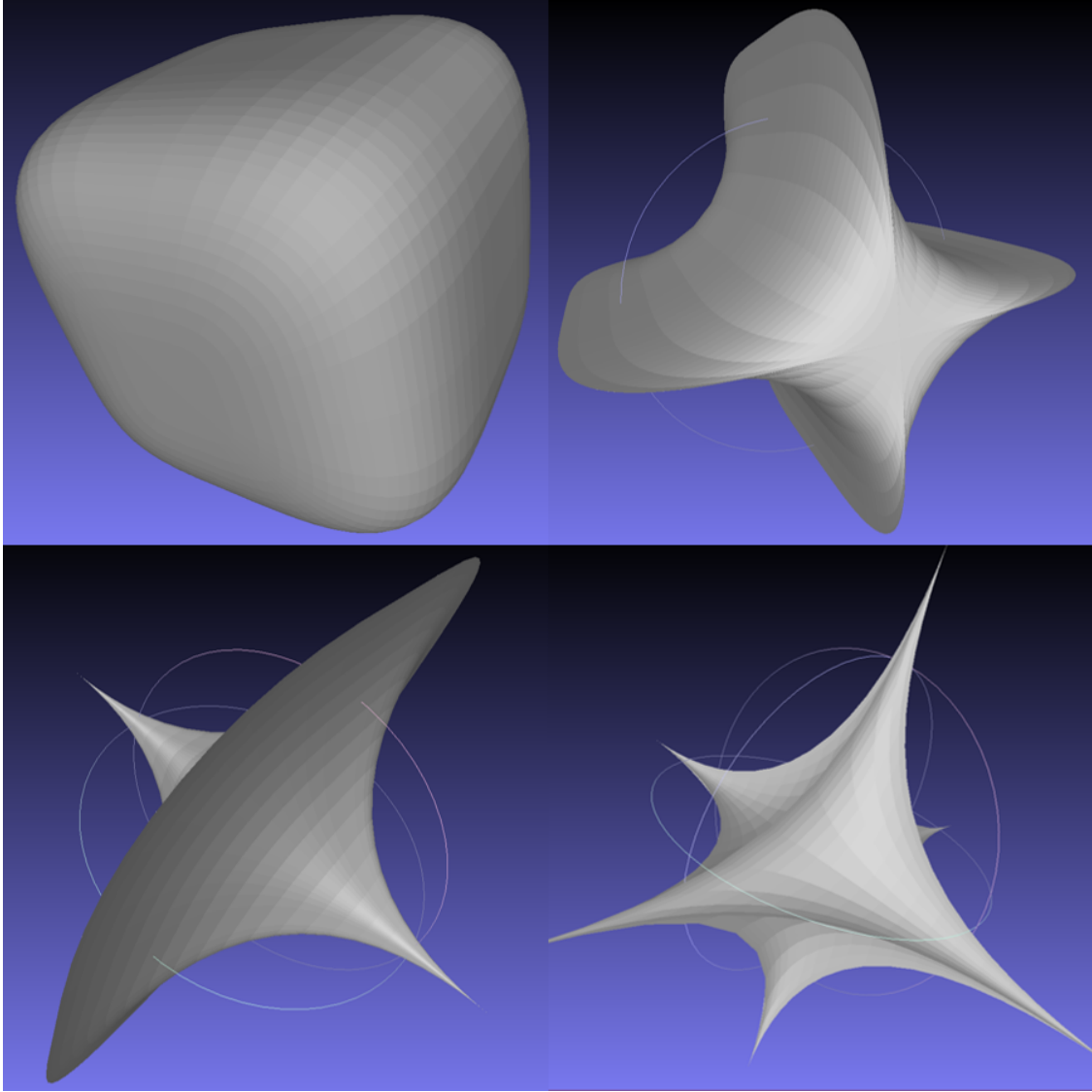


Figure 6.1: A subset of the VTK generated superellipsoid models. Parameters vary from the top left ($N_z, N_{xy} = 0$) to the bottom right ($N_z, N_{xy} = 4$). No regions with inverted surface normals are present.

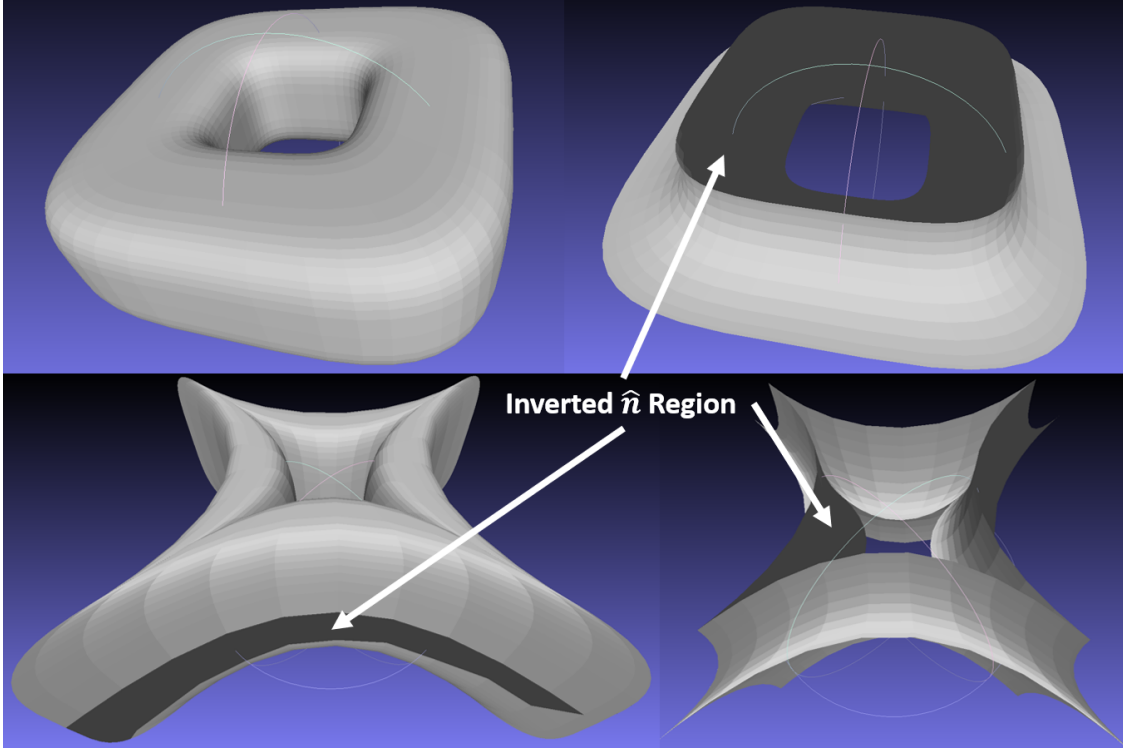


Figure 6.2: A subset of the VTK generated supertoroid models. Parameters vary from the top left ($N_z, N_{xy} = 0$) to the bottom right ($N_z, N_{xy} = 4$). The models also show inverted \hat{n} regions (dark gray).

Checking parametric surfaces for incorrect $T(i).\hat{n}$ (Algorithms 5, 3) used the previously determined thresholds, $\varepsilon_1 = 1e - 5$, $\varepsilon_2 = 1e - 8$ for mesh \hat{n} and point \hat{n} respectively. All superellipsoids passed both the mesh and point \hat{n} tests. All supertoroids passed the mesh \hat{n} test but seven of the eight with inverted \hat{n} regions generated operator warnings to visually inspect the mesh. The inverted \hat{n} region on the eighth model ($N_{xy}, N_z = 2, 0$) is symmetric and calculations balance out over the entire mesh. Therefore, the model passes the test. Thus, the supertoroid surfaces should be reconstructed before being TVF generation pipeline

input but testing did validate the testing for inverted $\hat{\mathbf{n}}$. While the pipeline, does not correct ill-formed inputs, it does inform the user when they are present.

During early trials, parametric surface task parameters were set to $d_{min} = 0.05$ m, $d_{max} = 1.05$ m, $d_{intra} = 0.5$ m, and $d_{inter} = 0.5$ m to provide three GVF layers [Sharp and Pryor, 2018]. These tests were repeated for comparison with test data before GVF layer resolution was checked and PCN evaluation where STL based task parameters are infeasible.

VF layers were calculated at distances from d_{min} to d_{max} with interpolation and voxelization applied to each layer only once. The output data reinforces the expectation of PCN growth is related to surface concavity (Figure 6.3). More convex models, $N_{xy} = N_z = 0$ (Figure 6.1, top left), show significant growth in the number of vertices with increasing distance but the most concave model, $N_{xy} = N_z = 4$ (Figure 6.1, bottom left), shows little growth (Figure 6.3). Supertoroid results are similar but highly erratic due to incorrect $T(i).\hat{\mathbf{n}}$ (Figure 6.3). The superellipsoid and supertoroid show significantly smaller layer sizes and more effectively display incorrect $T(i).\hat{\mathbf{n}}$ effects than early testing [Sharp and Pryor, 2018].

Checking the 18 superellipsoids and supertoroids VF layer resolutions provides several important observations (Figure 6.4). Superellipsoid resolution varies between models but is maintained between VF layers as expected. The erratic supertoroid results once again highlight the effect of inverted $\hat{\mathbf{n}}$. Superellipsoid and supertoroid VF layer sizes are smaller than previous results but the resolution is maintained. This suggests resolution was increasing significantly beyond task resolution, which could adversely impact mental load, in early investigations [Sharp and Pryor, 2018].

TVF graph structure construction fully connects VF layers and sparsely connects

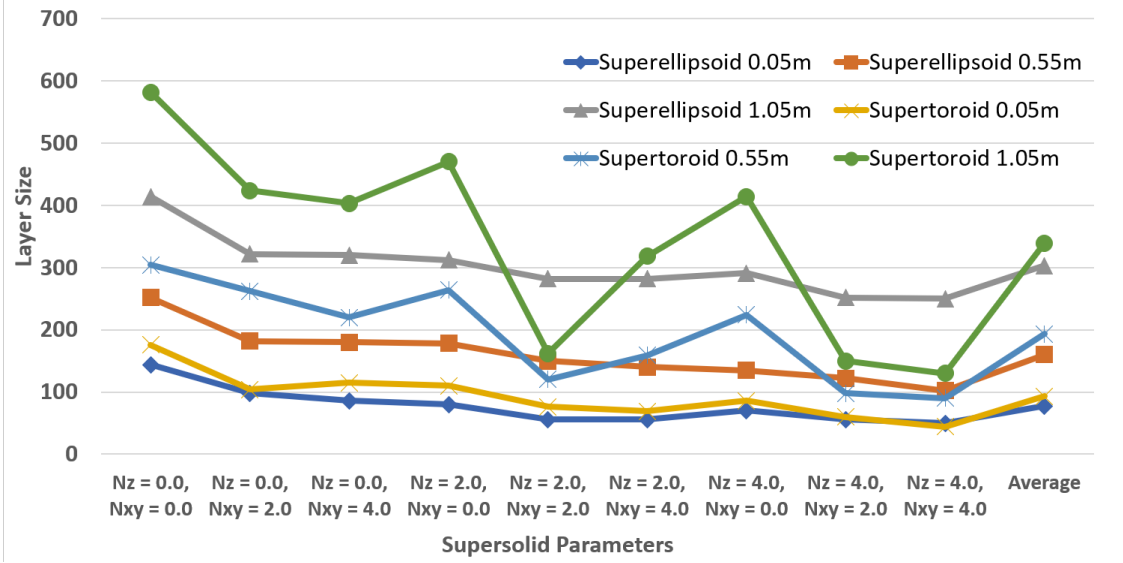


Figure 6.3: Graph of the number of VF layer vertices at distances of 0.05 m, 0.55 m, 1.05 m from the surface for superellipsoids and supertoroids. Supertoroid results are more erratic due to regions with incorrect surface normals.

between VF layers and examining intra- and interlayer edge growth displayed expected data trends. The number of intralayer edges (Figure 6.5) increases with the size of the VF layer (Equation 6.3). Interlayer edges increase more slowly and are approximately equal to the larger of the two linked VF layers (Figure 6.6). Thus, the generated VFs are intuitively maintained in reasonable resolutions for well-formed objects inputs into the generation pipeline.

$$E_{intralayer} = (N_{layer\ vertices}^2 - N_{layer\ vertices})/2 \quad (6.3)$$

The average of the average interlayer weights were calculated for each interlayer of the superellipsoid and supertoroids. All values were below set interlayer distance value, $d_{inter}=0.5$ m, as expected (Table: 6.1). The supertoroid averages were closer to 0.35 m and

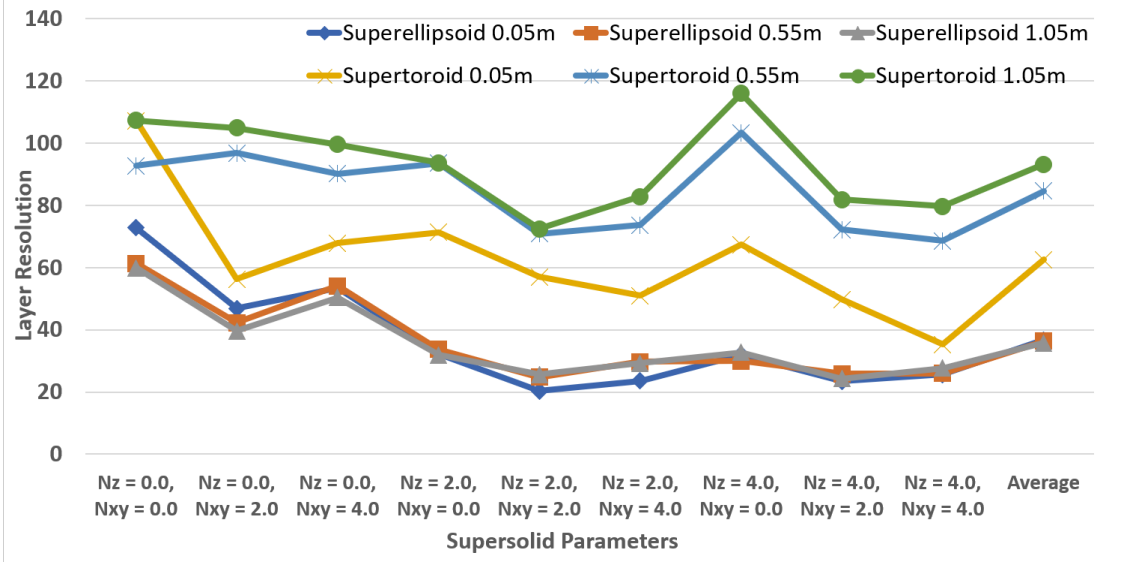


Figure 6.4: VF layer resolutions for three VF layers at distances of 0.05 m, 0.55 m, 1.05 m for both superellipsoids and supertoroids. Superellipsoid resolutions are highly similar but supertoroid results are more erratic due to regions with incorrect surface normals.

lower than the superellipsoid being due to the interior feature.

Table 6.1: Average interlayer distances averaged over all nine superellipsoid and supertoroids.

Superellipsoid Interlayer 1:	0.44 m
Superellipsoid Interlayer 2:	0.48 m
Supertoroid Interlayer 1:	0.39 m
Supertoroid Interlayer 2:	0.36 m

6.1.2 Polygonal Mesh Task Virtual Fixture Generation

The TVF generation pipeline was also evaluated on more complex polygonal meshes which were crowdsourced from Thingiverse [Thingiverse, 2018]. Eleven individuals each provided two to four models resulting in a pool of 32 models. Seven of the models were

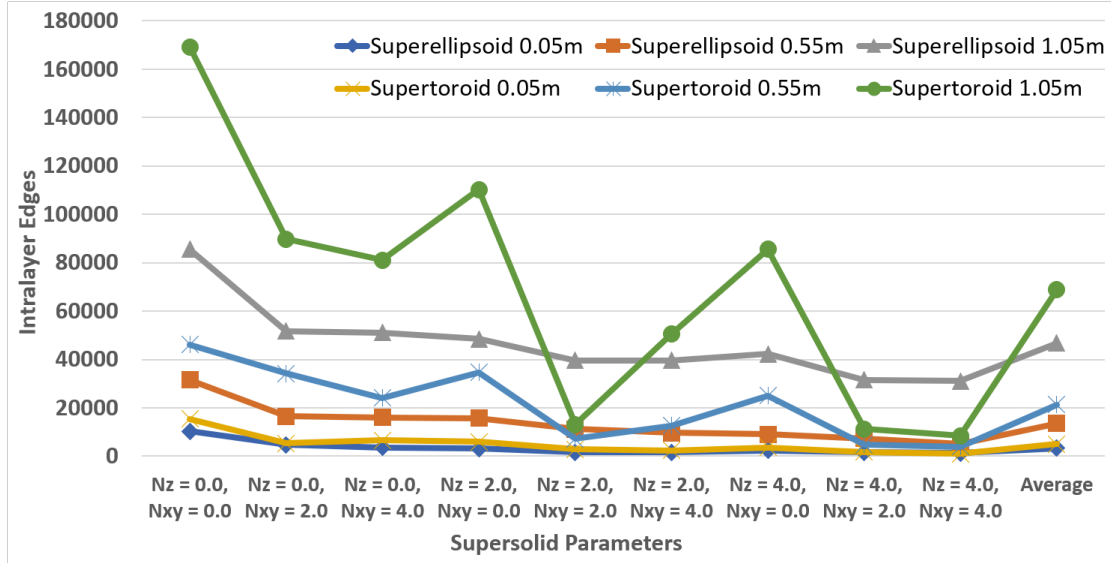


Figure 6.5: Intralayer graph edges at distances of 0.05 m, 0.50 m, 1.05 m for superellipsoids and superellipsoids. Supertoroid results are more erratic due to regions with incorrect surface normals.

eliminated as being unsuitable for batch testing due to containing multiple meshes or file size restrictions ($>5\text{MB}$). While TVF generation completes for large STL files, the resulting graph structure significantly taxes loading and visualizing in the MTTT environment (Figure 5.6). The remaining 25 models (Table 6.2) represent a large variety of items including cable holders (70549), desk figurines (906951), tools (1187995), and phone stands (2120591) (Appendix C).

As with parametric surface testing, point and mesh $\hat{\mathbf{n}}$ thresholds were set to $\varepsilon_1 = 1e-5$, $\varepsilon_2 = 1e-8$ respectively for mesh testing. Only one of the 25 models triggered an operator warning for either of the checks (3119803, mesh warning). Visual inspection found inverted $T(i).\hat{\mathbf{n}}$ on two holes (Figure 6.7) demonstrating the effectiveness of this straightforward technique for finding mesh inconsistencies.

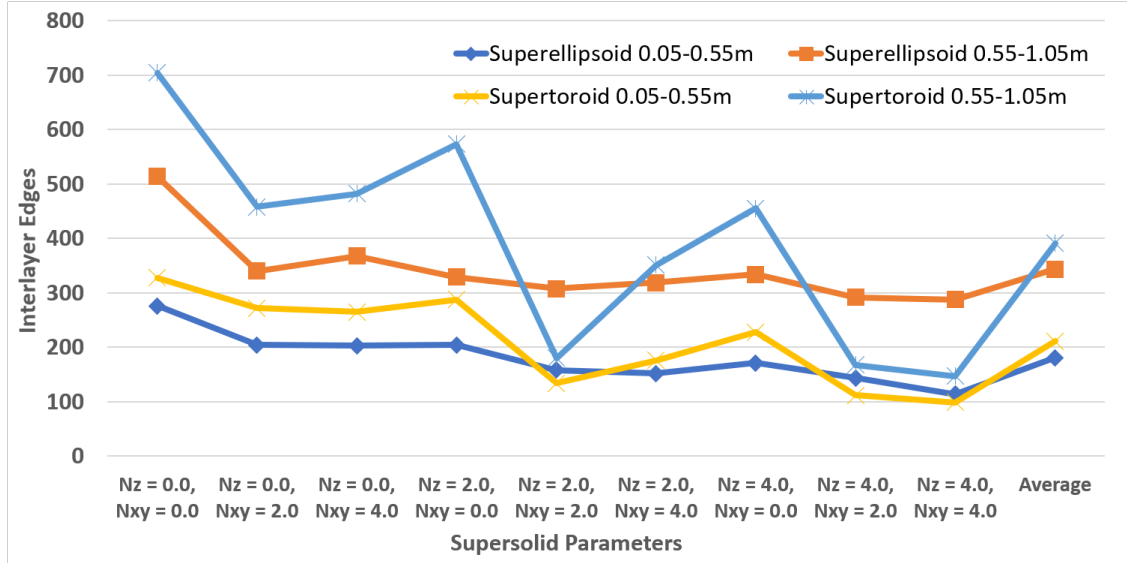


Figure 6.6: Interlayer graph edges between 0.05 m, 0.50 m, 1.05 m distances for superellipsoids and superellipsoids. Supertoroid results are more erratic due to regions with incorrect surface normals.

Table 6.2: Thingiverse models. Models are available in Appendix C and can be accessed through: "https://www.thingiverse.com/thing:<model_number>".

17314	38840	70549	906951	908062
1014845	1187995	1677784	2120591	3101067
3106129	3108035	3108554	3119494	3110862
3114718	3118241	3118847	3118855	3119665
3119580	3119670	3119735	3119802	3119803

Developing a general formula for assigning task parameters allowed comparison over a body of meshes with varying complexity (units, physical size, data size, etc.). Task parameters were based on the original STL average triangle side length, $\overline{T}(i)_{side}$, including the voxelization and resolution evaluation distance, r_{res} , (Equation 6.4). These values limited the graph file

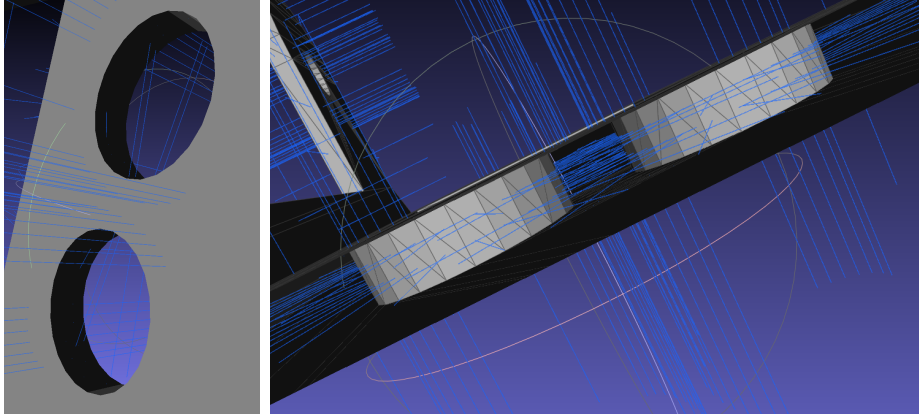


Figure 6.7: Thingiverse model 3119803’s interior holes with incorrect surface normals visualized in MeshLab [MeshLab, 2018]. Light gray surfaces have normals pointing toward the viewpoint while dark gray surfaces have normals pointing away from the viewpoint. The holes in the bracket are dark gray when viewed from outside the surface (left) and surface normals point toward the viewpoint when it is inside the bracket with dark gray interior surfaces above and below (right).

size while maintaining algorithm effectiveness. Another approach to the problem is to base task parameters on the largest model dimension. However, the dimensional approach is unable to account for the size of a model’s relevant features if they differ significantly from a model’s overall dimensions.

$$\begin{aligned}
 d_{intra} = d_{inter} = d_{min} &= 10 * \overline{T}(i)_{side} \\
 d_{max} &= 100 * \overline{T}(i)_{side} \\
 d_{voxel} &= d_{intra} \\
 r_{res} &= 3 * d_{intra}
 \end{aligned}
 \tag{6.4}$$

These parameters were tested on the parametric surfaces and three variations ($d_{intra} = 10 * \overline{T}(i)_{side}, 15 * \overline{T}(i)_{side}, 20 * \overline{T}(i)_{side}$) were applied to the polygonal mesh testing pool.

Average VF layer size grew at roughly an $N = C * dist^2$ rate, where N is the number of points and C is a constant (Figure 6.8). This result was expected since spherical surface area is proportional to $4 * \pi * r^2$.

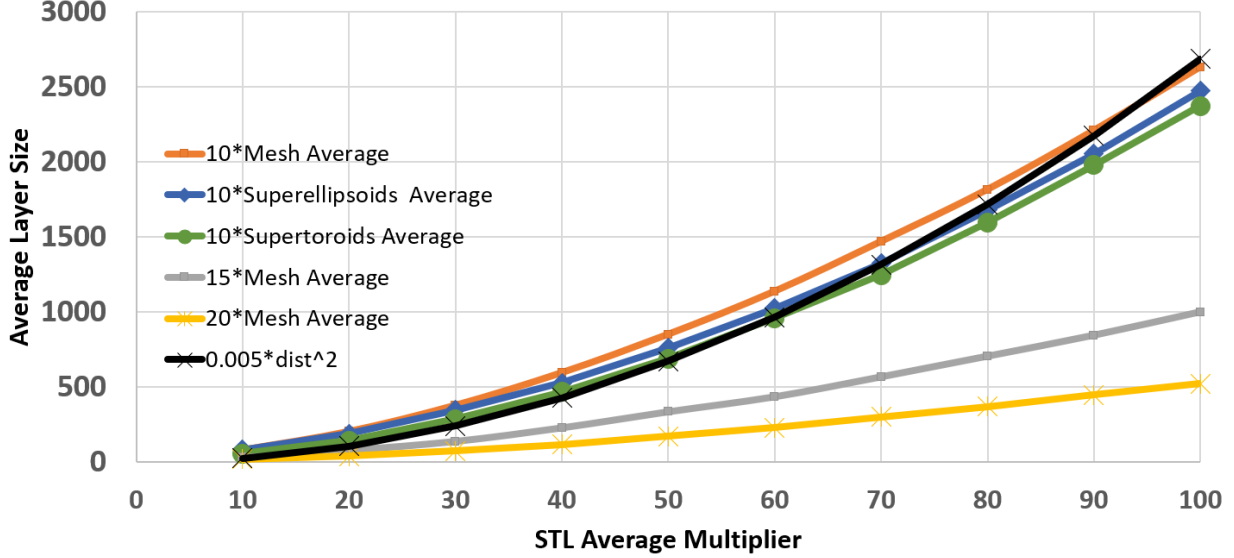


Figure 6.8: Parametric and polygonal mesh models evaluation results with varying intralayer distances. VF layer sizes were averaged per layer from 10 to 100 times $\bar{T}(i)_{side}$. Results show layer size growth rates of approximately $N = C * dist^2$, where N is the number of points and C is a constant. A comparison line of $0.005 * dist^2$ is displayed.

VF layer resolution evaluation used the same parameters. The results show a correlation between superellipsoids and polygonal meshes VF layer growth and resolution suggesting estimating task parameters based on $\bar{T}(i)_{side}$ is an effective approach. Average VF layer resolution does slowly decrease for all input types. Resolution decay at high d_{layer} to model size ratios could lead to cases where d_{intra} is unattainable. In such cases, authors recommend

higher task surface interpolation over multiple interpolation and voxelization iterations.

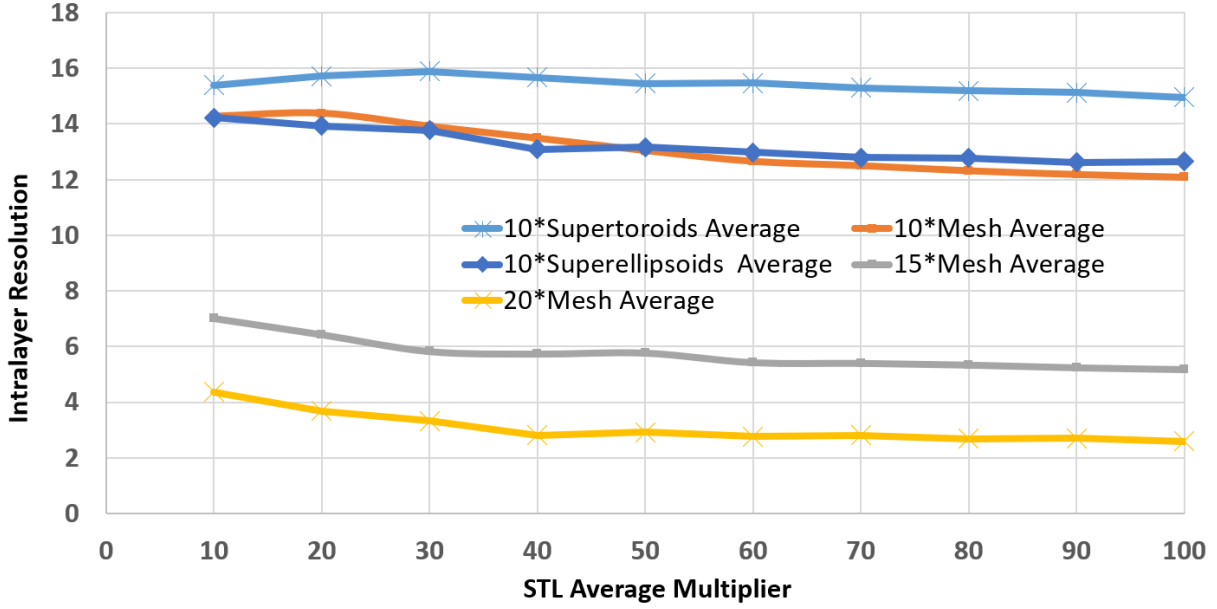


Figure 6.9: Parametric and polygonal mesh models evaluation resolution with varying intralayer distances. The r_{res} neighbors were averaged per layer for distances from 10 to 100 times $\bar{T}(i)_{side}$.

Four randomly selected models were chosen for additional investigation (Figure 6.10). Visual inspection of the first two GVF layers displayed the PCNs oriented back toward the surface at regular intervals. The first GVF layers are sparse, suggesting a lower d_{intra} might be necessary for task completion. Parameters are easily adjusted to task-specific TVF generation parameters from the generalized, model-based, parameters.

To verify interlayer distances the average interlayer distance was divided by the models d_{inter} to obtain a percentage. These percentages were then averaged for each of the first four interlayers among input categories (Table: 6.3). The percentages show the average interlayer

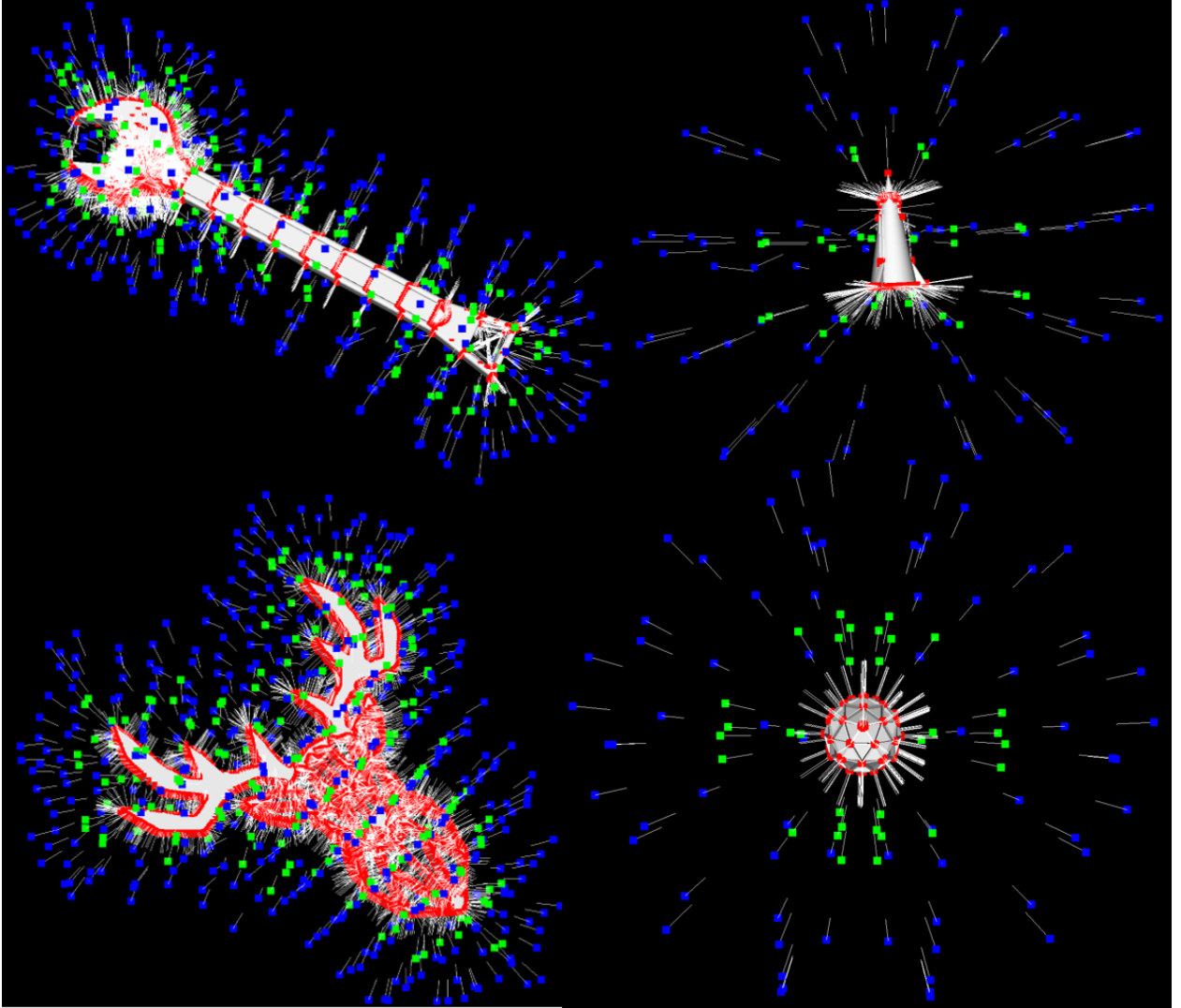


Figure 6.10: Thingiverse models wrench (1187995, top left), rocket (3101067, top right), antlers (3108035, bottom left), and icosahedron (3119665, bottom right) each with generated surface points (red dots), first (green dots), and second (blue dots) GVF layer points with point normals represented as white lines.

distances are below the mesh maximum, d_{inter} . Closer inspection reveals the superellipsoids have the highest percentage in every case but the supertoroids and polygonal meshes vary

more widely.

Table 6.3: Average interlayer distances percentage of d_{inter} averaged over all nine superellipsoid and supertoroids.

Superellipsoid Interlayer 1:	98.44 %
Superellipsoid Interlayer 2:	97.09 %
Superellipsoid Interlayer 3:	96.97 %
Superellipsoid Interlayer 4:	97.55 %
Supertoroid Interlayer 1:	83.27 %
Supertoroid Interlayer 2:	87.55 %
Supertoroid Interlayer 3:	85.91 %
Supertoroid Interlayer 4:	86.30 %
Polygonal Interlayer 1:	88.41 %
Polygonal Interlayer 2:	84.93 %
Polygonal Interlayer 3:	84.71 %
Polygonal Interlayer 4:	83.91 %

6.1.3 Point Cloud Task Virtual Fixture Generation

TVF generation was also demonstrated with point cloud with normals sensor data. 3D LIDAR data was gathered in a mock tunnel constructed at UT Austin to represent a nuclear facility exhaust tunnel (Figure 6.11, top) at Savannah River National Labs. A mobile manipulator is periodically used to perform methodical visual and radiation surveys [Pryor and Landsberger, 2017]. A section of the data around the large pipe was segmented for evaluation. Testing parameters varied from $d_{min} = 0.05m$ to $d_{max} = 1.05m$ and $d_{intra} = 0.1, 0.5$. Comparing results to superellipsoid and supertoroid calculations shows slower VF layer growth when $d_{intra} = 0.5$ due to the extension of a partial surface instead of a 3D object (Figure 6.12). Decreasing d_{intra} raises the layer size above superellipsoid and supertoroid levels.



Figure 6.11: TVF generation pipeline tested on PCN data taken at UT Austin’s mock of the H-canyon tunnel [Pryor and Landsberger, 2017] (top).

Virtually inserting a Yaskawa SIA20, based on previous use, into MTTT interface provides data visualization, reachability information, and EEF motion around the TVF (Figure 6.13, bottom). OMPL [Sucan et al., 2012] and *MoveIt!* [Sucan and Chitta, 2017a] perform IK and planning scene checks respectively but could be replaced with alternate solvers due to the modular nature of ROS. Thus, TVFs are applicable to a wide range of robot applications including emergency response, remote exploration, etc.

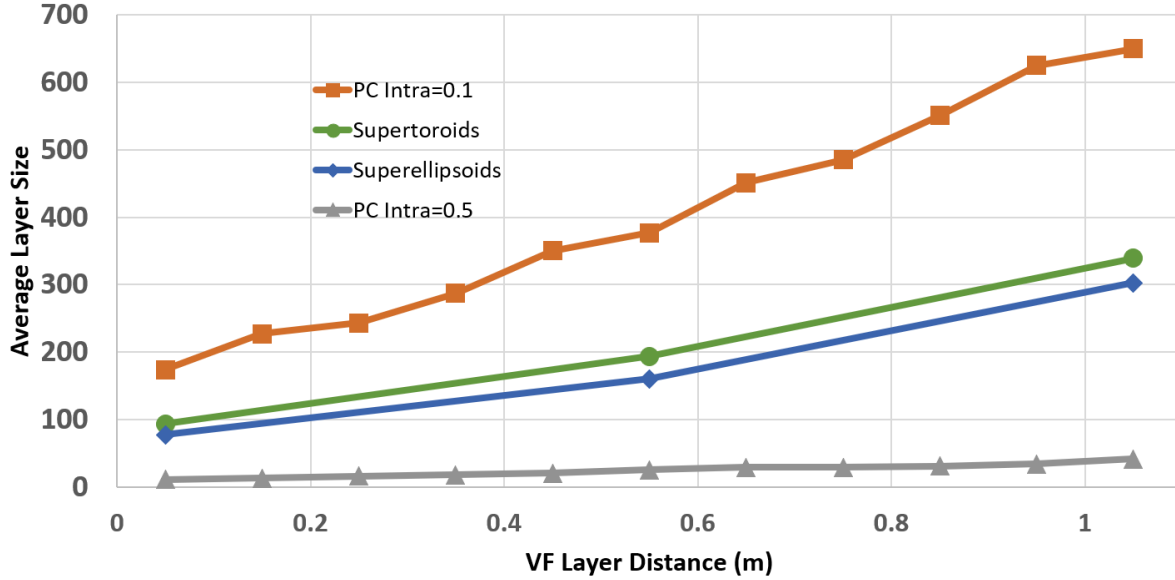


Figure 6.12: Point cloud data evaluation from 0.05 m to 1.05 m with varied intralayer distances 0.1 m and 0.5 m. VF layer sizes are compared to supersolid testing data.

6.1.4 Summary of Task Virtual Fixture Generation for General Surfaces

This section presents TVF generation was evaluated on data from multiple sources. Parametric surface and polygonal mesh input testing resulted in several detected incorrect $T(i).\hat{n}$. However, one defect was missed due to symmetry in the incorrect $T(i).\hat{n}$ regions. Task surface and parameters are used to generate layers of offset interpolated and voxelized point clouds which can be used as a FRVF or GVF. A TVF graph is constructed from the GVF layers and provides a more expressive VF than volumetric primitive approaches. TVF generation was completed for the parametric surfaces, crowdsourced polygonal meshes, and PCN sensor data. In each data case intralayer resolution slowly decreases with increasing distance from the task surface due to geometric expansion. If the resolution decreases below

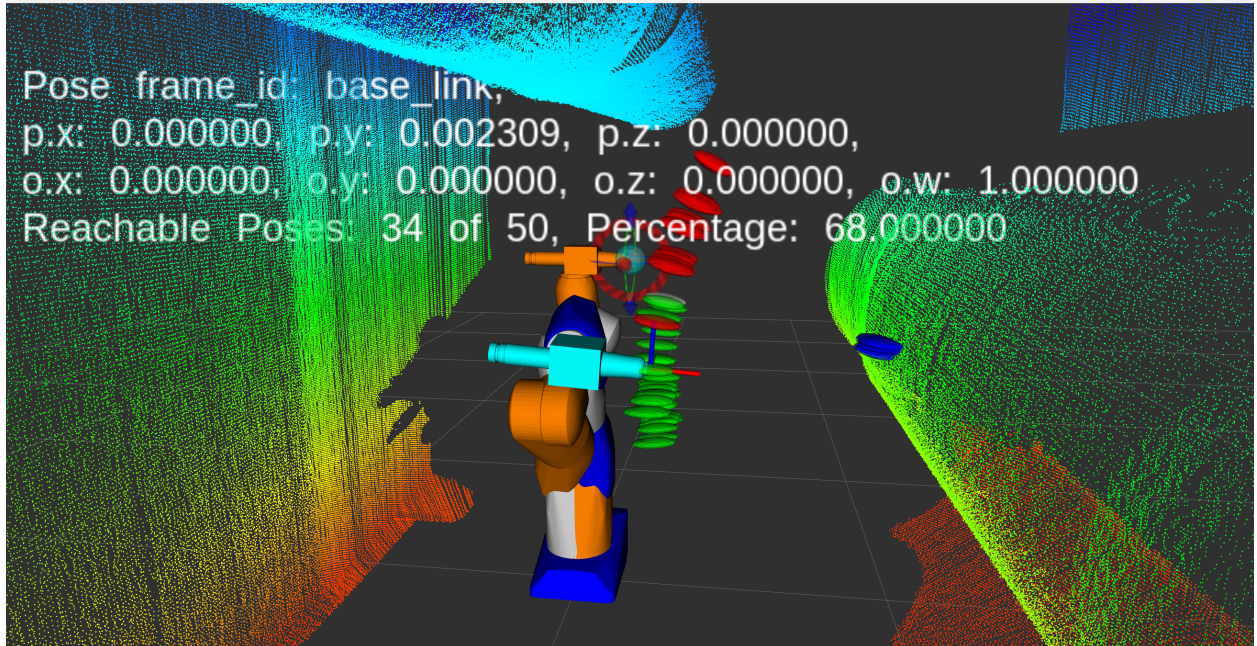


Figure 6.13: The results are visualized using the MTTT. A SIA20 previously used in this research thread for D&D tasks replaced the NRG VaultBot which gathered the data. The interface provides reachability feedback. Red: current GVF layer but unreachable. Green: current GVF layer but reachable. White: current pose in TVF. Blue: pose in layer closer to or farther from the task surface.

task requirements the authors recommend higher interpolation of the task surface instead of the GVF layer. These results were deemed a success and research proceeded on to operator evaluation. At this stage it was noted large TVF graphs can strain the MTTT environment resulting in some practical limitations on task parameters and input mesh size.

6.2 Operator Task Virtual Fixture Evaluation

While the previous section outlines mathematical evaluation of the TVF generation output the goal of this research is to provide assistance to operators. In order to conduct

research with human participants Institutional Review Board (IRB) approval is required. Therefore, an application for expedited approval was submitted. The application contained a section for both spatially discrete and spatially continuous testing. Expedited approval was applicable since ‘the study involves no more than minimal risk’ [United States Department of Health and Human Services, 2018]. Approval was received based on safety procedures at place at LANL and those outlined in the application, UT Austin IRB 2018-06-0092. All of these documents are provided in Appendix D.

6.2.1 Discretized Task Virtual Fixture Operator Evaluation

Experiments were necessary to evaluate the interpretability of TVFs and usefulness of the MTTT for spatially discrete non-contact tasks. In order to test reachability, a manipulator model was placed in the virtual *RViz* environment. The Yakasawa SIA20 manipulator with ROS-I [ROS-Industrial, 2015a] integration was chosen (Figure 6.14). Previous VNSVF research on D & D tasks used the SIA20 due to increased reach and payload [Sharp et al., 2018]. The EEF is an IPG Photonics Compact Cutting Head [Photonics, 2018] laser cutter (Figure 6.14). The cutting head was chosen based on parameters for D & D tasks [Hilton and Khan, 2014]. Next, task parameters were altered from the previous research in order to decrease GVF layer size and facilitate rapid planning scene checks for an entire GVF layer. The appropriate GVF layer size was determined to be approximately 50 poses which kept reachability calculation time below 10 sec (Figure 6.14).

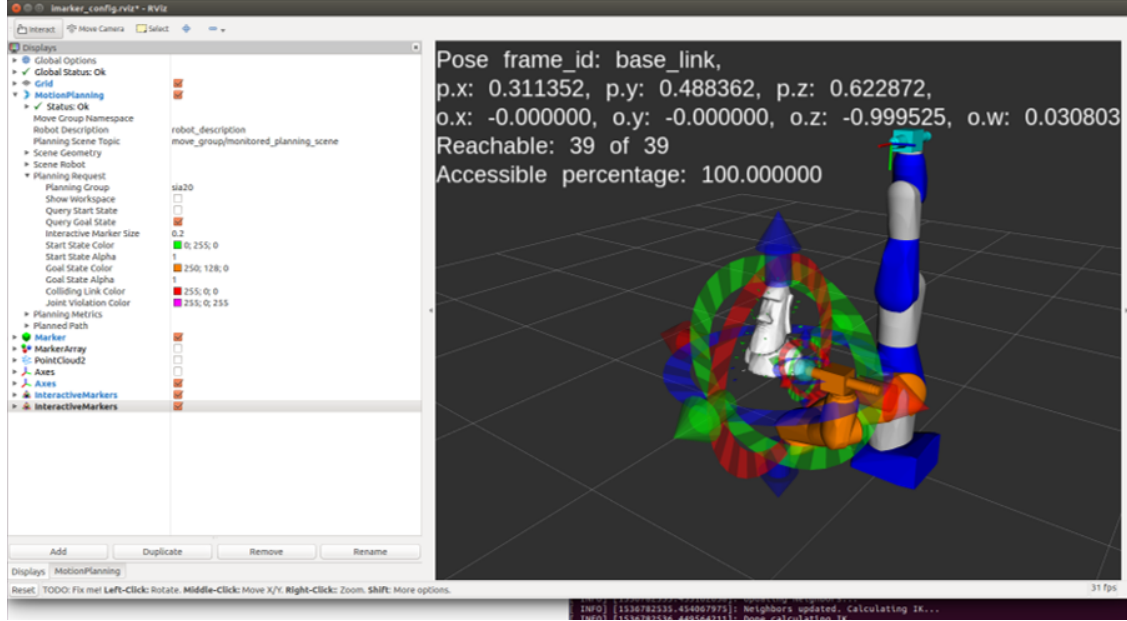


Figure 6.14: MTTT interface showing interactive markers for task FRVF and SIA20 manipulator control. Pose and reachability feedback is summarized and displayed for the current task FRVF location. Red: current GVF layer but unreachable. Green: current GVF layer but reachable. White: current pose in TVF. Blue: pose in layer closer to or farther from the task surface.

6.2.1.1 Task Surface Test Set Selection

The task surface test set for this operator evaluation is a subset of the parametric surfaces (Figure 6.15, right) mathematically evaluated in Section 6.1.1 [Sharp and Pryor, 2018]. The surfaces were generated using Visualization Toolkit (VTK) [The Visualization Toolkit, 2018] in the parameter space $0 \leq N_z, N_{xy} \leq 4$ at intervals of 2 (Eqn. 6.1). The supertoroids tested in Section 6.1.1 were removed from the operator evaluation set to avoid incorrect surface normals. The task surface set also includes a model from the crowdsourced set of Thingiverse [Thingiverse, 2018] models (Figure 6.15, left) mathematically tested in

Section 6.1.2. The Moai or stone monolith contains both convex and concave surfaces but no large interior regions. The subset of models used were compatible with the capabilities of the laser cutting system and thus most relevant to the primary D & D user base.

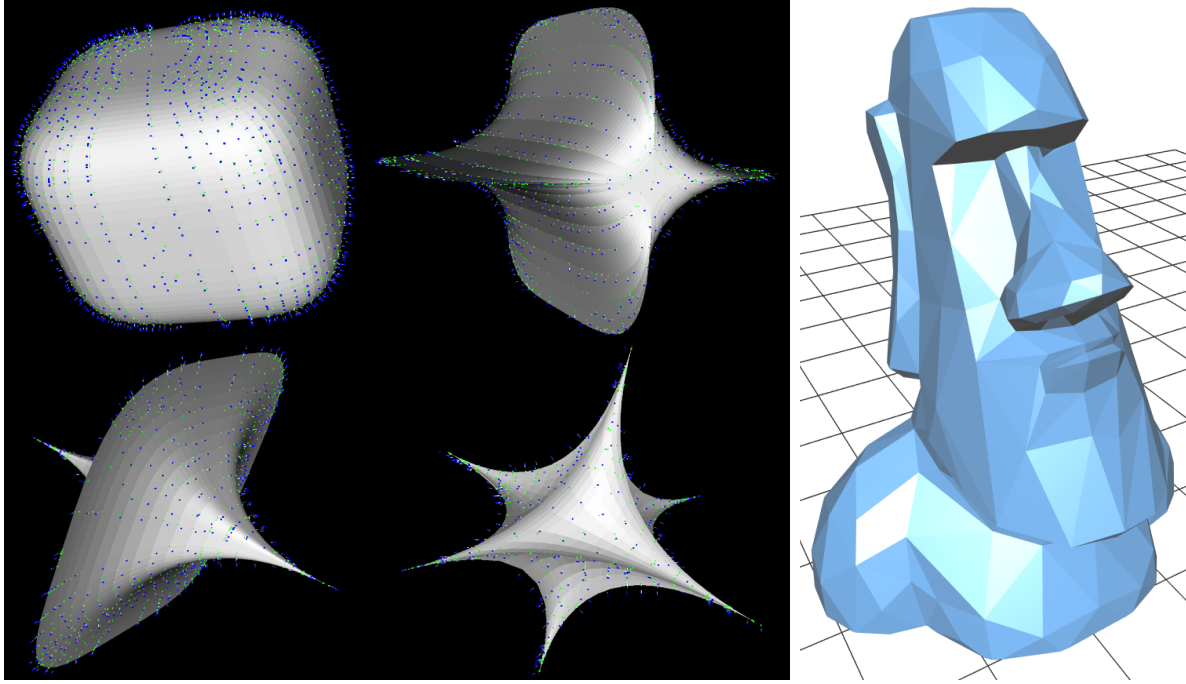


Figure 6.15: A subset of the VTK generated superellipsoid models. Parameters vary from the top left ($N_z, N_{xy} = 0$) to the bottom right ($N_z, N_{xy} = 4$). The models also show the surface points (green dots), their surface normal (white lines), and the first TVF layer (blue dots) (left) [Sharp and Pryor, 2018]. Thingiverse Moai monolith model (<https://www.thingiverse.com/thing:908062>) (right).

6.2.1.2 Experimental Procedure

Volunteer subjects were shown the MTTT interface (Figure 6.14) and read a test script describing the interface, testing procedure, and recorded data (Appendix E). They were asked to rate their experience with robotic manipulators on a scale of one (no experience)

to ten (a self-described robotics expert). A one-ten scale was selected to assure sufficient fidelity in the presence of floor / ceiling effects [Cindy Passmore et al., 2002]. They were then allowed to explore the simulated environment and interface for up to five minutes. The exploration environment was loaded with the $N_z = 4, N_{xy} = 4$ superellipsoid model (Figure 6.15, left side bottom right). Test subjects were allowed to view the TVF pose reachability for this model before it was removed for the tests described below.

Users were divided into two tracks (T1, T2). Each track included four models and four trials for each model. The order of the first three superellipsoid models were randomized and the Moai monolith model was always the final experiment. For the first and second T1 trials, users were provided with the task model and asked to place it in a location where they thought the robot would be able to reach the largest portion of the task surface. They were provided with the only reachable percentage of the task surface as feedback. Users were then asked to choose a location with higher reachability based on the provided feedback. Once the first two trials were completed, users were asked to complete a Likert scale questionnaire (Table 6.4). For the third and fourth trials, users were allowed to examine individual TVF vertex reachability in addition to being provided the reachability percentage (Figure 6.14). After completing the third and fourth trials, users completed another Likert scale questionnaire (Table 6.4).

In the second user testing track, T2, users were allowed to examine individual TVF pose reachability and were provided the reachability percentage during all four trials with each of the four models. Users filled out the Likert scale questionnaire (Table 6.4) after the second and fourth trials. After the fourth trial on the Moai monolith model, users were allowed to continue searching for locations with higher reachability. For this test TVF pose

Table 6.4: Spatially discrete Likert scale questionnaire

	N/A	Strongly Disagree (1)	Disagree (2)	Neutral (3)	Agree (4)	Strongly Agree (5)
This was a difficult task						
This was a frustrating task						
The interface was easy to use						
I successfully completed the task						
I would improve with practice						
I was unsure where to place the part						

reachability feedback was not shut off in order to move the object thus providing immediate feedback to the user and increasing the search rate.

6.2.1.3 Operator Evaluation Results

The user group consisted of eleven participants with varied levels of robot experience (Table 6.5) and distributed between track one (T1) and track two (T2) to maintain test diversity.

Table 6.5: Test Subject Robotic Manipulator Experience Levels

Experience	1	2	3	4	5	6	7	8	9	10
T1 subjects	1	1	0	1	1	1	0	1	0	0
T2 subjects	1	0	1	0	0	1	1	1	0	0

Four models were evaluated by each participant using the procedure detailed above. All but one user opted to try immediate feedback with the Moai monolith model. Therefore, 185 tests were preformed. Results from the four trials with each of the four models were averaged among T1 and T2 (Figure 6.16). The results show users took longer to choose their location when they began without TVF pose reachability information even when it was provided in trials three and four (Figure 6.16). Users also achieved a higher reachability when feedback was provided first even after the final trial with the Moai monolith model (Figure 6.16).

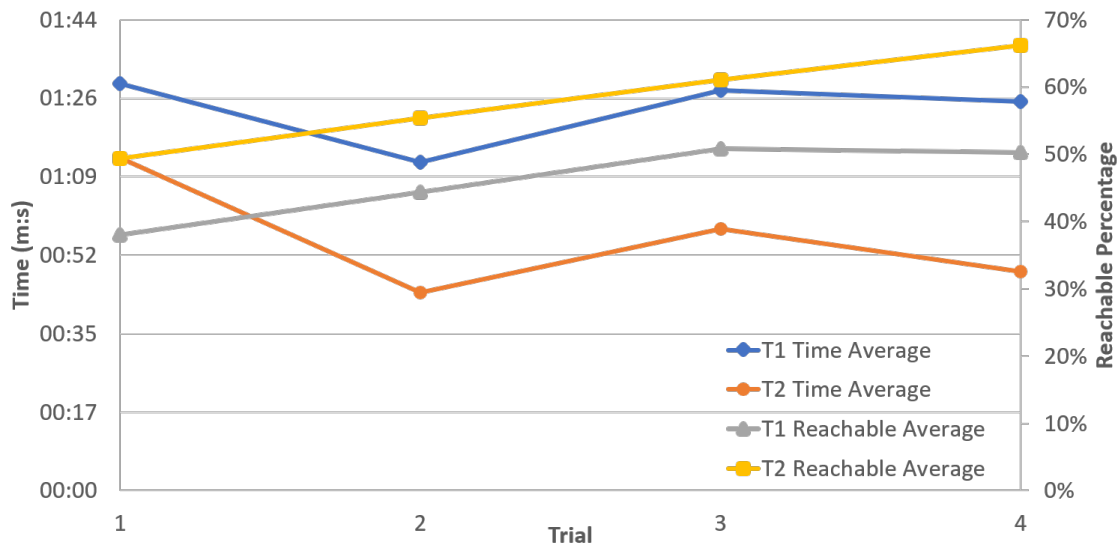


Figure 6.16: T1 and T2 averaged trial time (left) and reachable percentage (right) for the four trials.

Data from the Likert scale questionnaires was also aggregated based on T1 and T2 tracks and the difference between the two was calculated (Figure 6.17). The results show users thought the task was more difficult and frustrating without TVF vertex reachability

information even for complex shapes. The usability between the T1 and T2 user interfaces was extremely flat with a difference of only -0.01. Users also had higher agreement that they were more likely to improve with practice and successfully completed the task on T2 compared to T1. Results from the last questionnaire inquiry demonstrate a greater certainty in task surface placement when TVF vertex reachability information was provided.

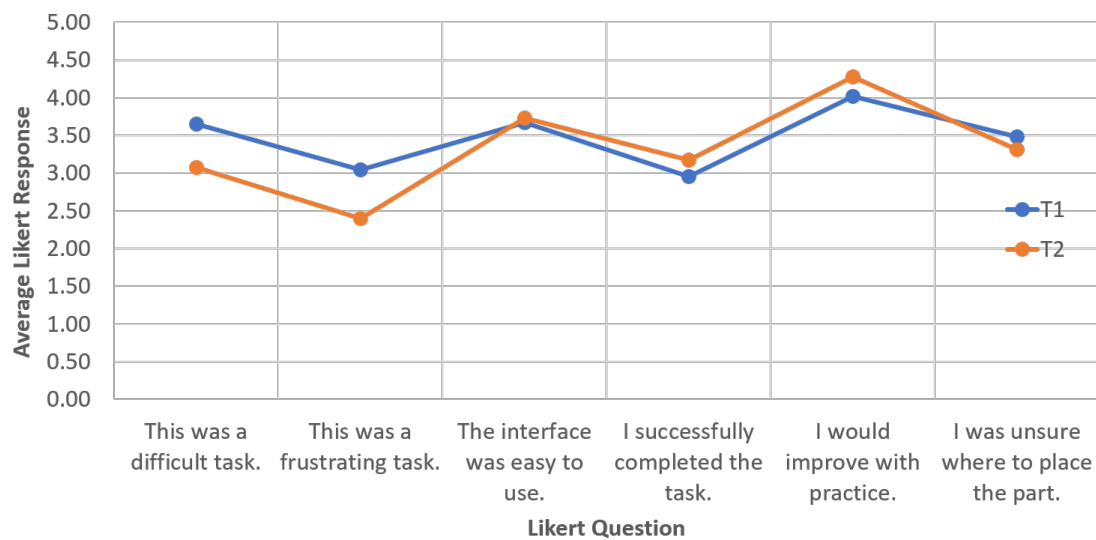


Figure 6.17: T1 and T2 averaged Likert scale responses (Table 6.4).

Additional insights were gleaned from the operator comments and direct experiment observation. MTTT reachability information (Figure 6.14) use varied but with a consistent pattern across all users. The least and most experienced users were less likely to take advantage of the TVF reachability information. Inexperienced users were slow to recognize it's value, and experienced users instead utilized a well-developed mental model, having previously worked with kinematically similar manipulators. Users with some but limited experience used the MTTT more extensively. Additionally, several users asked when they

would get TVF vertex reachability information back once they had completed a trial where it was available. This is consistent with customer interview practices recognizing a latent need [Cindy Passmore et al., 2002]. Other users wanted 'live' feedback before it was provided to them in the optional final trial. Many users were impatient with lag caused by reachability analysis in a variety of trials. These observations held no matter the task surface model order.

6.2.1.4 Summary of Spatially Discrete Task Operator Evaluation

This section examined operator feedback for TVF use with the MTTT on spatially discrete non-contact tasks. It demonstrates TVFs generated from complex geometries are still interpretable to operators. This is the case even with the lack of a discernible directional control structure which was utilized previously with VNSVFs. Users were asked to maximize the reachable percentage of the task surface for several models. The model test results show a higher reachability was achieved in less time when the TVF was visualized. Likert questionnaire results support this data suggesting visual feedback resulted in task which was a less difficult, less frustrating, and had an increased future improvement likelihood. Therefore, it seems acceptable to conclude TVFs are still operator interpretable when paired with the MTTT for spatially discrete non-contact tasks. TVFs and the MTTT can therefore be used for multiple applications including reachability analysis and constrained tele-operation around complex objects. Also, the MTTT can also be used for training, task setup, or even robot hardware design.

6.2.2 Continuous Task Virtual Fixture Operator Evaluation

Additional operator evaluation experiments were performed to evaluate the interpretability of TVFs for spatially continuous non-contact tasks on industrial manipulator hardware. Previous user testing with LOAs used post task surveys [Sharp and Pryor, 2015; Sharp et al., 2017a] but for this evaluation something more extensive was required. The experiments were designed to yield similar comparative results to those of [Bruemmer et al., 2002; Kruusamae and Pryor, 2016]. Operator evaluation also follows some of the guidelines proposed by [Chiou et al., 2015] which include:

- Tasks must require teleoperation and autonomy to be better or successfully completed.
- Extensive participant training makes experiments time consuming but ensures trust in the autonomous system.
- Situational awareness affects performance but is very complex to measure.
- Operator workload is difficult to measure in real time without using physiological techniques.
- Degraded performance, jointly with context, might be simplified by using experiments with ‘idle time’ as a performance metric.

Due to uncertainty during joint reconfigurations with *Descartes*, an alternative interface to the MTTT was used for spatially continuous task operator evaluation. Thus, TVF vertex poses were integrated into ABB’s proprietary RobotStudio software [ABB, 2018b] as outlined in Section 5.2.2. This interface allowed virtual testing and safe hardware execution of spatially continuous task paths. The ABB hardware systems at LANL are contained within highly static workcells meaning situational awareness required a limited amount of mental load.

6.2.2.1 Task Description and Surface Material Selection

The task selected for operator evaluation was plasma cleaning. Plasma cleaning is a process in which a surface is exposed to a plasma to clean or treat it. Cleaning may include the removal of organic compounds and treatments can create chemically stable layers on a surface. Surfaces are exposed to the plasma inside of a chamber or with a small surface area applicator. For operator evaluation the surface exposure type is a small applicator referred to as a plasma pen (Figure 6.18, on top of box).



Figure 6.18: PVATePla PlasmaPen Atmospheric Plasma System [PVATePla, 2018].

This task was selected due to a the tight positional tolerances required for effective

plasma pen operation. The plasma pen available for these experiments was a PVATePla PlasmaPen Atmospheric Plasma System (Figure 6.18). It uses a single phase, four amp max power supply. The plasma pen can use a variety of input gases including compressed air, N_2 , N_2/H_2 , and O_2 among others. The treatment band width is 3 mm to 10 mm depending on the desired treatment [PVATePla, 2018]. The tight tolerances required by the task meant it is better completed with autonomy and, therefore, fulfills the first guideline set by [Chiou et al., 2015].

6.2.2.2 Task Surface Material and Geometry Selection

The goal of the operator evaluation task material was to be sensitive to the plasma pen output plasma and could be produced in a variety of shapes. Therefore, the 3D printed Stratasys Vero PureWhite photopolymer [stratasys, 2018] was chosen for operator evaluation due to compatibility with the available 3D printers. Material discoloration by the plasma pen was dependent on the pen travel speed and the distance to the material surface.

Once the task material was determined a task geometry was required. Geometry selection criteria included having geometric complexity, being asymmetric, and originating from a CAD model. The task geometry chosen for operator evaluation is a model of a flag waving in the wind. It was designed by a machinist to test the capabilities of a new machine. The flag model is asymmetric and contains multiple convex and concave regions of varying degrees. It was scaled down to limit task setup / executions time but also still allow hardware execution which was limited by the plasma pen treatment band size (Figure 6.19).

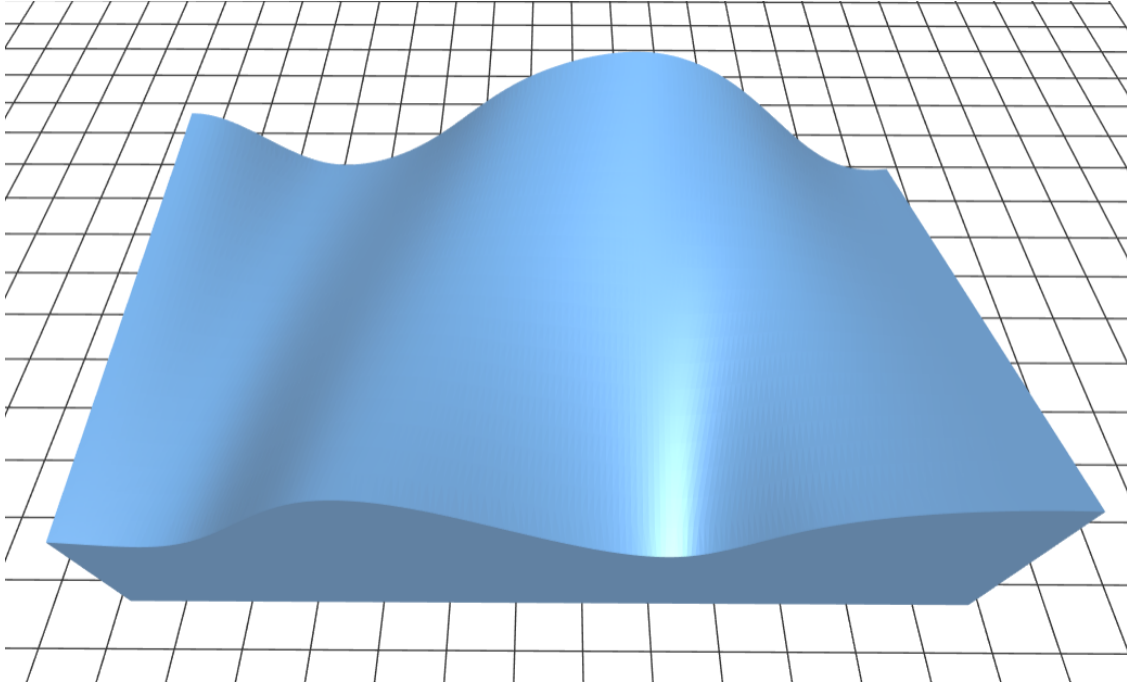


Figure 6.19: Complex and asymmetric model of flag waving in the wind.

6.2.2.3 Manipulator Hardware and Operator Selection

There were three industrial manipulators available at LANL for operator testing. Two of these manipulators were a Motoman SIA5 and SIA10 with payloads of 5 kg and 10 kg, respectively (Figure 6.20). These 7 DOF manipulators allow greater flexibility in motion planning and increase the likelihood of generating trajectories which are unintuitive to the operator. Another industrial manipulator is the ABB IRB 140. It is a 6 DOF manipulator with similar reach (810 mm) and payload (6 kg) to the Yakasawa SAI5 [ABB, 2018a]. While these SIAs have more complete ROS driver integration, through the FS100 controller, the decision to separate spatially continuous task testing from *Descartes* made this fact irrelevant. Since more possible operators had been trained on ABB software and hardware than Motoman,

the IRB 140 was chosen for spatially continuous evaluations. The IRB 140 was also function in a LANL workcell (Figure 6.21) next to a plasma pen gas source and was one of the available ABB training manipulators.

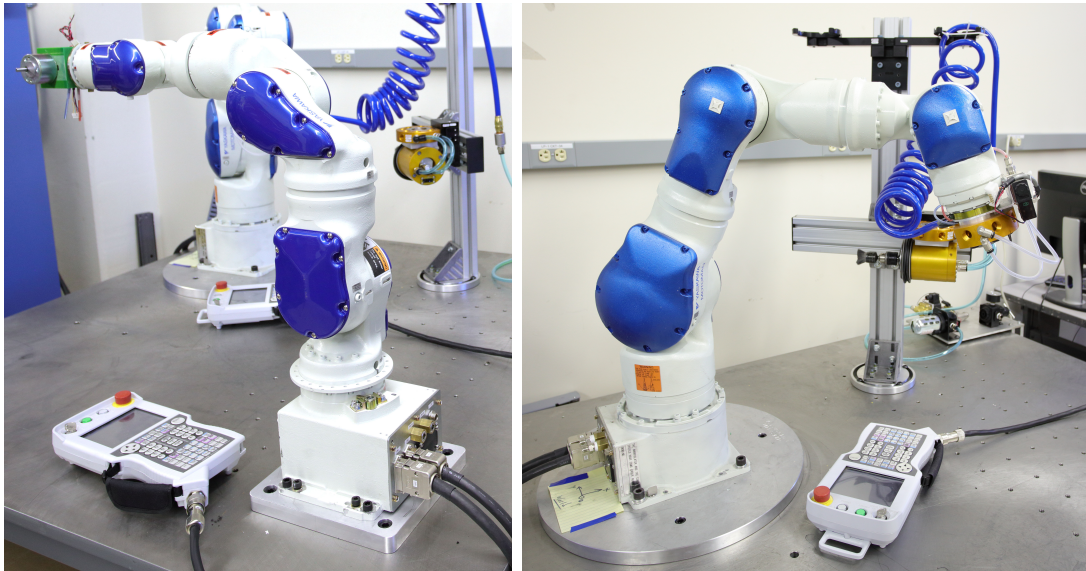


Figure 6.20: The Motoman SIA5 (left) and SIA10 (right) 7 DOF manipulators with tooling from previous projects at LANL.

Due to the training requirement for test subjects to use ABB software and hardware only six individuals had the required training. The Primary Investigator and an advisor were two of the six individuals and were removed to avoid research bias. The other four test subject had different levels of ABB training but had all completed the first class which was sufficient to proceed with TVF evaluation. Setting an operator requirement of having ABB training is both practical for using proprietary software and ensures trust in the hardware system satisfying the second guideline set by [Chiou et al., 2015].



Figure 6.21: ABB IRB 140 chosen for TVF hardware operator evaluation.

6.2.2.4 Experimental Setup

The experimental setup consisted of an IRB 140 mounted to an aluminum table with the work space on an elevated table next to it (Figure 6.22, bottom). The test piece was further elevated off of the table next to the robot with machining blocks (Figure 6.22, bottom center). The task surface was placed on machining blocks in order to avoid EEF collisions with the wrist while cleaning the task surface region closest to the manipulator base. The

task surface was also placed in the positive X and negative Y quadrant of the manipulator workspace to ensure the entire task surface was accessible without joint reconfigurations. These steps were taken by the Primary Investigator (PI) to simplify path construction during setup and minimize operator concerns during path execution. Normally, such checks would be performed by the operator based on their degree of manipulator experience.

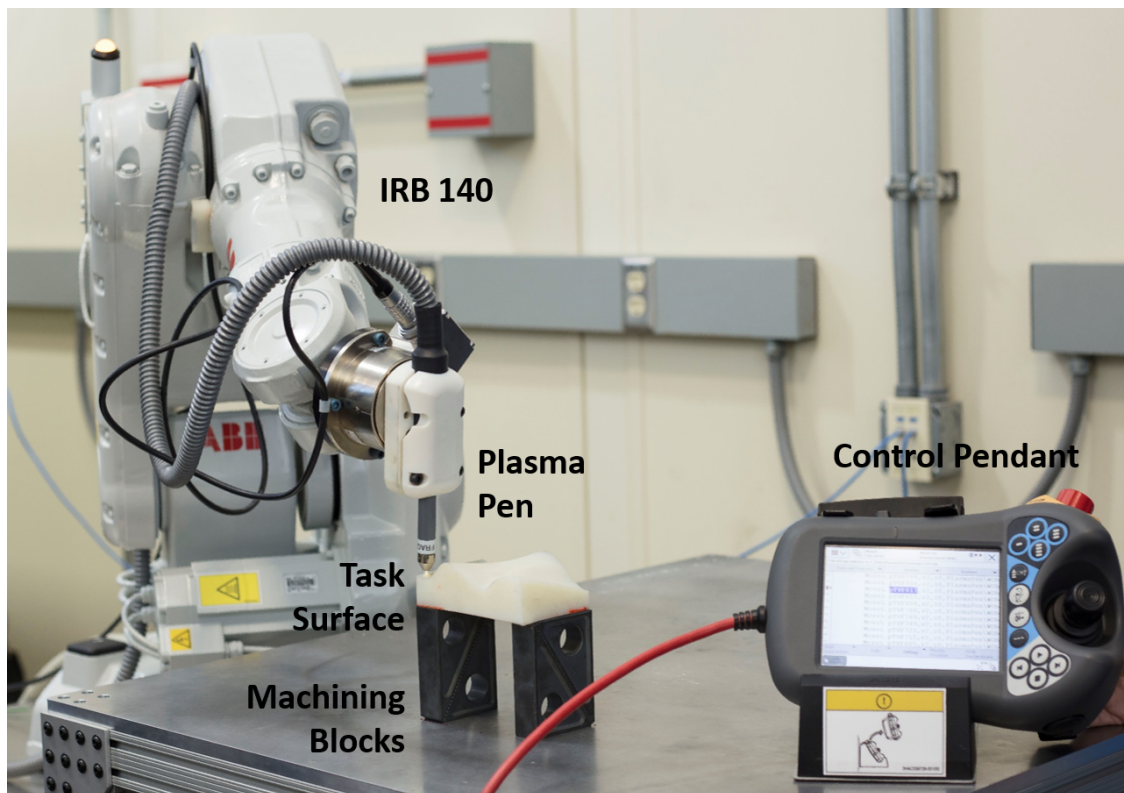


Figure 6.22: ABB IRB 140 hardware setup for TVF operator evaluation with the elevated workspace platform (bottom) and the test surface raised on machining blocks (bottom center).

The input gas for the plasma pen for these experiments was compressed air since it is safe and readily available. Other gases may have provided a more effective plasma but plasma

effectiveness was beyond the scope of these experiments. The plasma pen was mounted to the IRB 140 with a 3D printed compression fitting (Figure 6.22, center). As a safety precaution the mount allowed the plasma pen to slide along its axis, back through the mount, when exposed to contact forces above a threshold. Plasma pen control was connected to one of the ABB's digital outputs. Therefore, it could be controlled either through the control box switches, the ABB pendant, or programmatically.

Another safety precaution in place during the setup phase was to use a test piece with the same geometry as the task surface but 3D printed in a flexible material. The flexible material allowed operators to test locations and paths with lower collision concerns but was less susceptible to plasma discoloration.

Before beginning trials, the operators were read a script outlining the trial procedure and recorded metrics (Appendix G). They were also given an overview of RobotStudio by the PI's advisor who has additional training. Operators were also allowed time to refresh their ABB programming skills as a group in order to create a more uniform test pool.

Two task execution methods were compared during operator evaluation. The first method uses the standard steps for creating robot poses which is taught in ABB training. Robot poses are recorded by moving the robot to the desired location using the ABB control pendant (Figure 6.22, bottom right). The EFF pose, robot configuration, and external axis data are then recorded with a pose ID in the RobTarget format (Listing 5.3). Pose information is recorded in the selected work object frame allowing the work object transform to be changed but the robot poses to remain correct. Unique poses are created individually but when transforms are known, such as with linear surface coverage paths for a planar surface, poses can be created and edited in RobotStudio. Once the necessary poses are

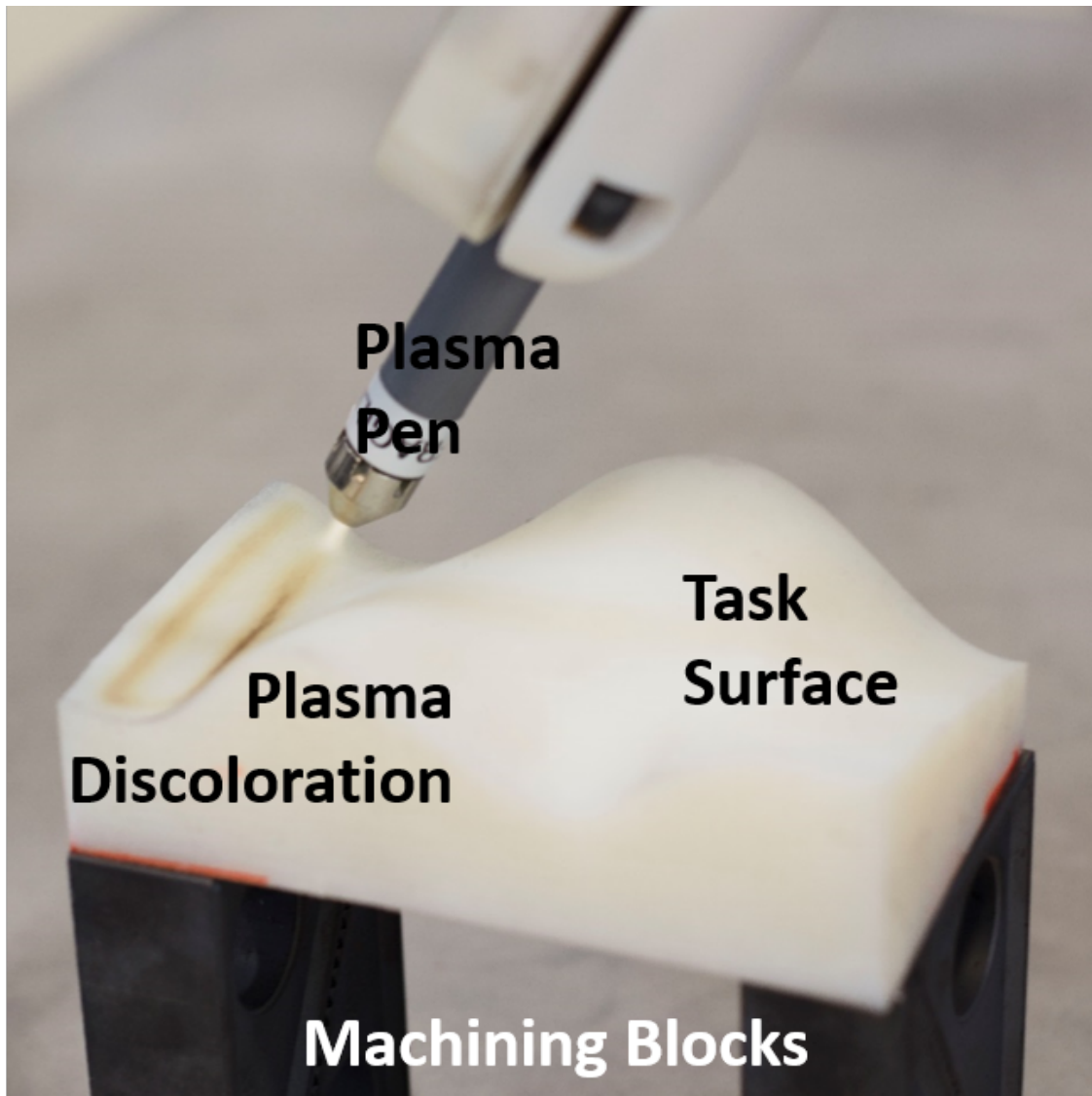


Figure 6.23: Close up of TVF operator evaluation plasma pen and task surface.

created the task path is constructed using joint or linear manipulator moves. Paths can also be constructed while creating poses when overall task execution is known. This approach

is common as it allows operators to gain a greater sense of the motion between poses than creating a path from pre-generated poses.

The second task execution method allows operators to use TVF poses to construct the task path. GVF layer poses (Figure 6.24) were converted from point clouds into RobTargets and visualized in RobotStudio as outlined in Section 5.2.2 (Figure 6.25). The use of TVF poses alleviates the need to manually create each unique task pose. However, poses in addition to the TVF poses can be created is desired. Therefore, TVF poses can be integrated into overall automation where there might be multiple task surfaces or multiple tasks on a single surface.

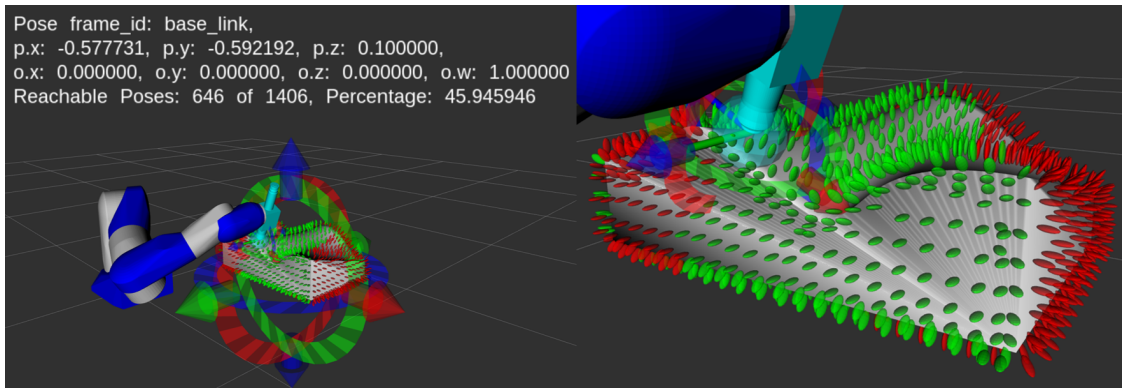


Figure 6.24: MTTT representation of the spatially continuous testing surface with reachability analysis for the Yaskawa SIA20.

The first test method for each operator was randomly assigned to reduce test bias caused by increased manipulator use. Thus, operators two and four used the ABB standard first and the other two users started with TVF poses available. Each operator also performed two trials with TVF poses available. The difference between these trials was the task specified intralayer distance. In the lower resolution test the intralayer distance was 6 mm and in

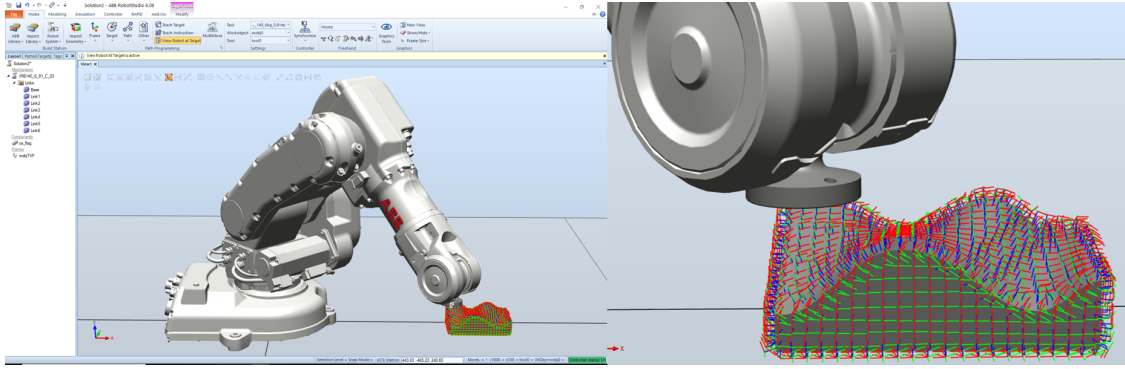


Figure 6.25: RobotStudio interface for spatially continuous testing with the IRB140 (left) and a closeup of the TVF frames (right).

the higher resolution test the intralayer distance was 3 mm. These two experiments were performed in order to gain qualitative feedback on the effects of resolution on operator and task performance.

6.2.2.5 Quantitative Evaluation Results

Quantitative feedback was collected in multiple forms. First, operator setup times, execution times, and the number of collisions were measured. Setup data displays a very wide range of times for ABB standard task setup. The range however decreases for low resolution TVFs and tightens further for high resolution TVFs (Figure 6.26). The average ABB standard setup time was quadruple the low resolution TVF test and almost double the high resolution TVF test. Operator three declined to perform the low resolution TVF pose test and were excluded from setup time average calculations.

The average execution time approximately doubled between the manual and high resolution TVF tests. Execution times maintained a similar range through the tests but

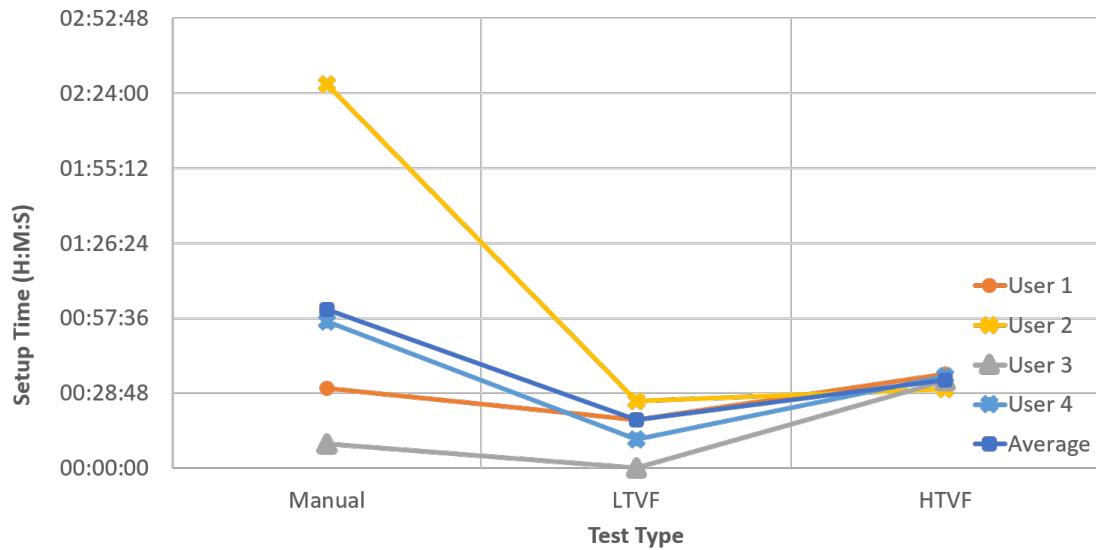


Figure 6.26: Spatially continuous testing setup time data for all four operators and the average.

operator switched locations in the spread. Operator three declined to perform the low resolution TVF pose test and were excluded from execution time average calculations.

Pictures of testing samples were also gathered for analysis (Figure 6.28). Evaluation bins pixels into unclean, clean, and over clean categories based on RGB values. If a pixel's difference between the red and blue values was below 50 the tone had not yellowed and the pixel was considered unclean (Figure 6.28 top left, lower half of sample). If the red value dropped below 100 the pixel was considered over clean, effectively burned, (Figure 6.28 top left, top). Other pixels were considered clean (Figure 6.28 bottom right). This method allowed approximation of the amount of surface cleaned during the trials while also providing some information on the distance from the plasma pen to the surface (Figure 6.29). Since the test speed was set, except in operator three's second test where it was purposefully exceeded,

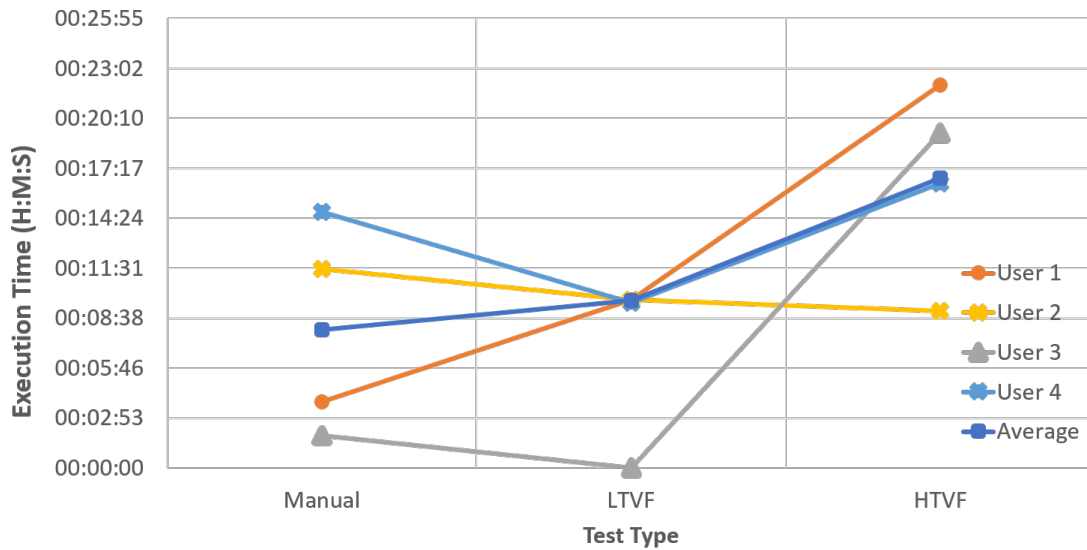


Figure 6.27: Spatially continuous testing execution time data for all four operators and the average.

increasing or decreasing the plasma pen distance to the surface would leave it unclean or over clean respectively.

Analysis of the pixel based results (Figure 6.29) demonstrated a higher clean surface percentage for low resolution and high resolution TVFs (Figure 6.30). The over clean percentage is highest for manual assignment but remains below 1%. These results correlate to qualitative visual inspection of the binned pixel images (Figure 6.29).

To provide a more succinct visualization of the spatially continuous qualitative testing, pixel analysis results were divided by the setup time to provide and approximate pixels clean per minute of setup time. This data demonstrates average cleaning rates with low resolution TVF and high resolution TVF poses are approximately four and three times higher, respectively (Figure 6.31). Cleaning rates are higher for low resolution tests suggesting the

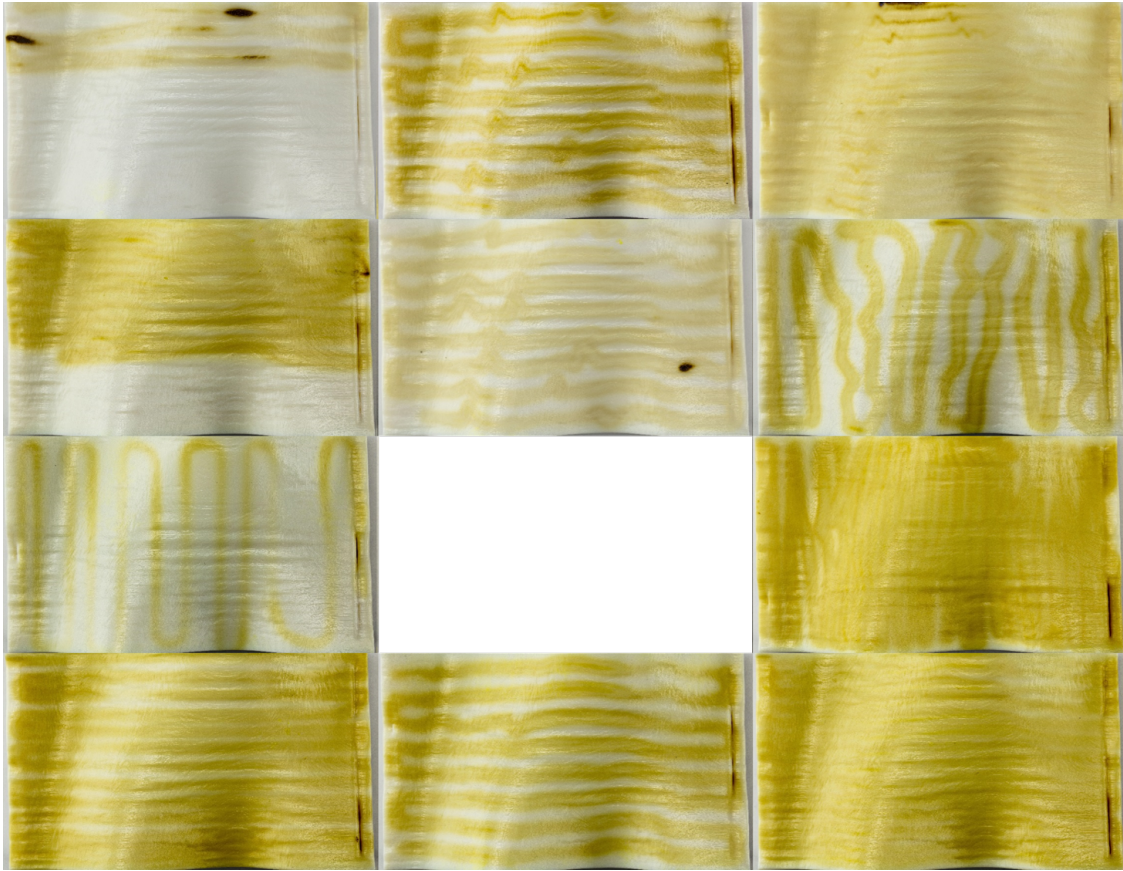


Figure 6.28: Spatially continuous plasma testing raw data for manual (left), low resolution TVF (middle), high resolution TVF (right) tests for all four operators. The gap represents and incomplete test.

interface became visually crowded at higher resolutions. The lowest cleaning rate (User 2, manual assignment) also corresponds the longest test (2 hours and 27 minutes) and the highest number of collisions, ten. The only other test with collisions was User 1's manual trial where there were two collisions with the task surface which also corresponds to the second lowest cleaning rate. There were no collisions when TVF poses were available. This suggests a significant safety increase when performing a task with pregenerated TVF poses

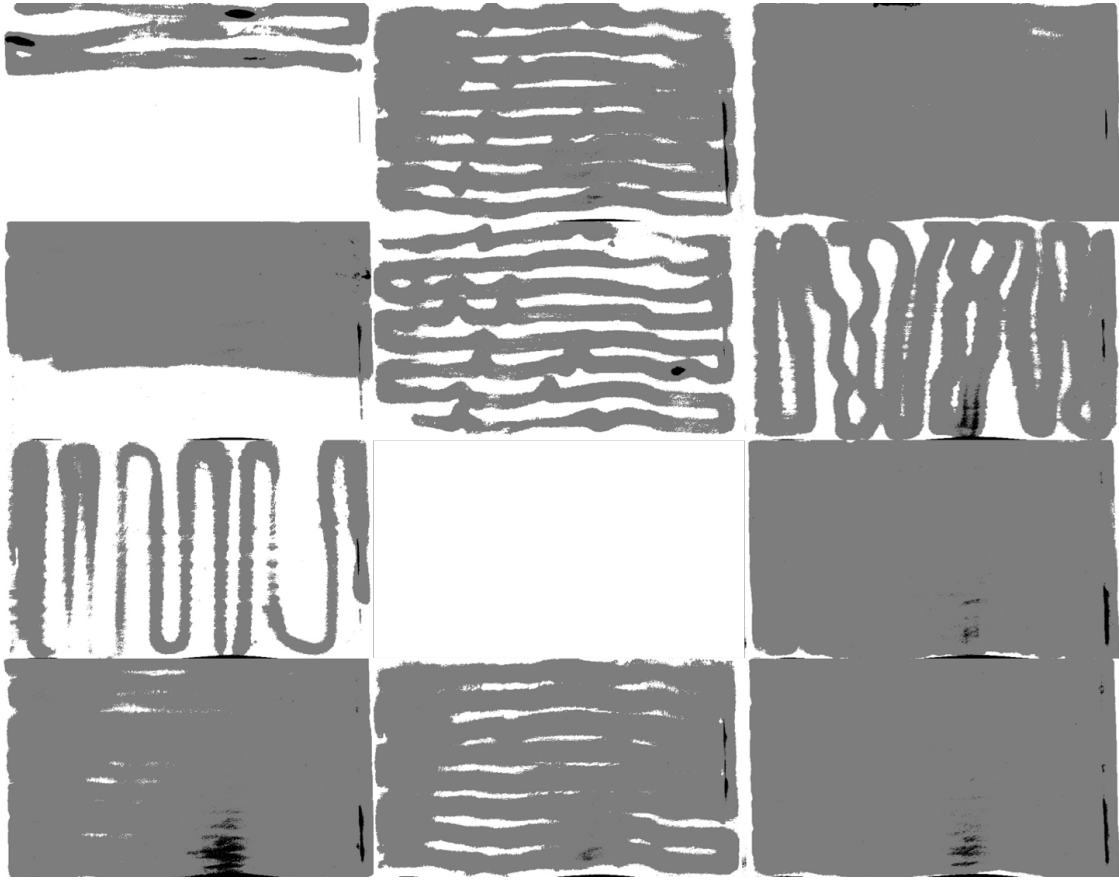


Figure 6.29: Spatially continuous plasma testing analyzed data for manual (left), low resolution TVF (middle), high resolution TVF (right) tests for all four operators. The gap represents and incomplete test. Pixel data was binned into unclean (white), clean (gray), and over clean (black).

over manually assignment. User three achieved the highest manual assignment cleaning rate but cleaned only 41 % of the task surface as seen in the visual data (Figure 6.28 third row, first column). User four achieved the highest cleaning overall but this was also their final trial (Figure 6.31).

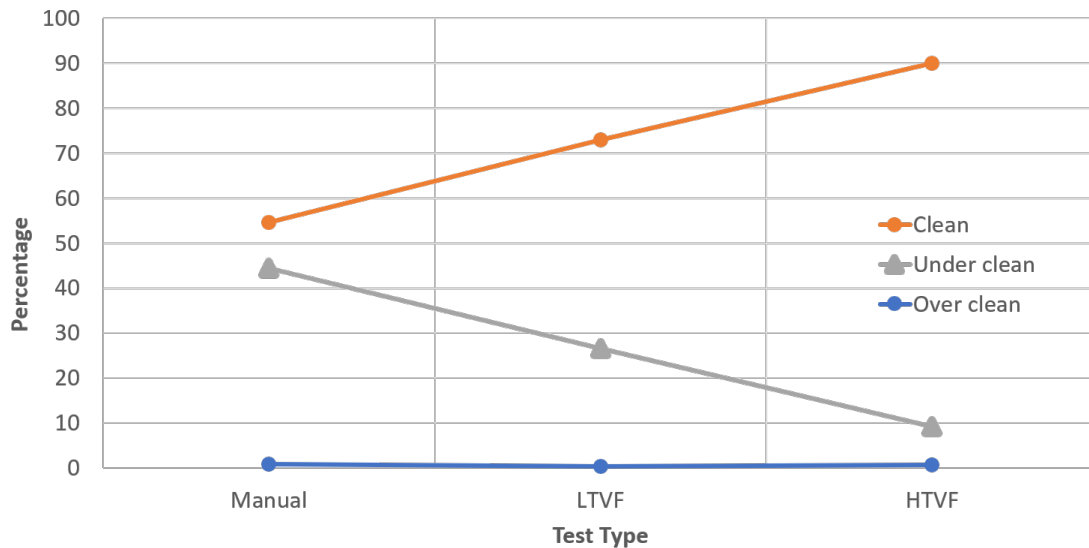


Figure 6.30: Spatially continuous testing unclean, clean, and over clean surface percentages with manual, low resolution TVF poses, and high resolution TVF poses.

6.2.2.6 Qualitative Evaluation Results

Qualitative feedback was collected in two forms. First, operators filled out a Likert scale survey after each trial (Table 6.6). Operator comments and PI observations were also collected during experiments.

Questionnaire results were averaged over manual assignment, low resolution TVF, and high resolution TVF testing (Figure 6.32). The results show some expected and unexpected trends. The average interface usability was highest with low resolution TVF poses and the same between manual assignment and high resolution TVF poses. This result suggests the interfaces with TVF poses are at least as easy to use as an industrial interface. Results also show a much more significant relationship between task frustration, difficulty, and success. When TVF poses are available the task is less frustrating, less difficult, and more successfully

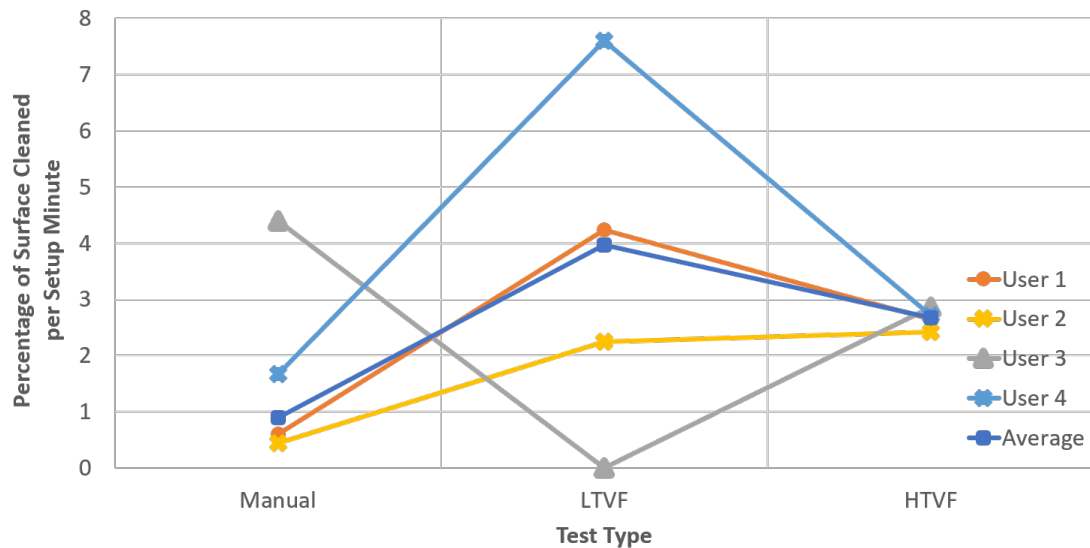


Figure 6.31: Percentage of surface cleaned per minute of setup time for spatially continuous testing with manual, low resolution TVF poses, and high resolution TVF poses.

completed.

Due to different interpretations and testing order the value of the next Likert question is unclear (Figure 6.32). None of the operators manually assigned poses during the low resolution trial and therefore all answered N/A to manual assignment time commitment. In other cases users answered N/A to the effects of TVF poses when performing manual assignment first. The results still suggest having TVF poses increased setup speed, process success, and operators would prefer to have TVF poses available in the future.

A large portion of operator comments discussed various tedious elements of the testing. Such comments included the manually assigning poses, selecting many TVF pose to form the path, and the speed of task execution. User 4's first trial was manual assignment and achieved the highest manual score at 94 % clean, commented, "The process would have taken much

Table 6.6: Spatially continuous Likert scale questionnaire

	N/A	Strongly Disagree (1)	Disagree (2)	Neutral (3)	Agree (4)	Strongly Agree (5)
The interface was easy to use						
This was a frustrating task						
This was a difficult task						
I successfully completed the task						
Manually assigning poses was the most time consuming portion of the task						
Having TVF poses available increased setup speed						
Having TVF poses available increased process success						
I would prefer to have TVF poses available for task setup in the future						

longer and been more frustrating if I had taken orientation into account”. This approach was effective due to the spherical formation of the plasma but would be ineffective for many other tasks. Comments on the tedium of manually assigning poses were more common once the operator had preformed a trial with TVF poses available. For example, User 1 commented, “I already don’t want to do this anymore” after less than 10 min and “Can I be done now, this sucks. I’ll do one more row.” after less than 25 min working on the manual test. For reference, User 1 had low resolution TVF poses first and completed the task in under 20 min.

Comments related to selecting TVF poses mainly occurred right after a “mis-click” incident where the operator accidentally clicked on the task surface and needed to restart the

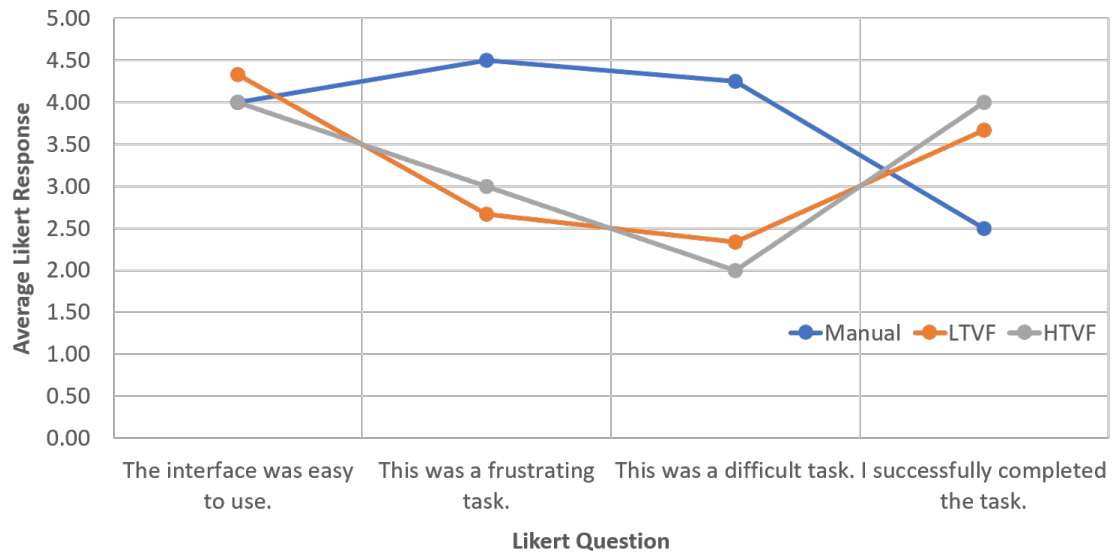


Figure 6.32: Average responses for Likert scale questions one through four for manual assignment, low resolution TVF, and high resolution TVF testing.

section of path they had been selecting. For example, User 2 commented, “I kept losing the points I was going to add to my path” and “It would be helpful if you couldn’t click on the task surface.” The likelihood of “mis-click” events increases with the number of TVF poses necessary to construct the path and was therefore more likely during high resolution TVF pose trials. This lead to the User 1 comment “As frustrating as it is to ‘mis-click’ it is less frustrating than manually assigning poses. It is an error within my control and I can fix it by being more patient.”

There were more observations than comments on the speed of task execution but one example came from User2, “You could cover more ground with a bigger pen”. Observations include that in all trials were TVF poses were available operators watched only a portion of the simulation and hardware execution before moving on the execution with the solid test

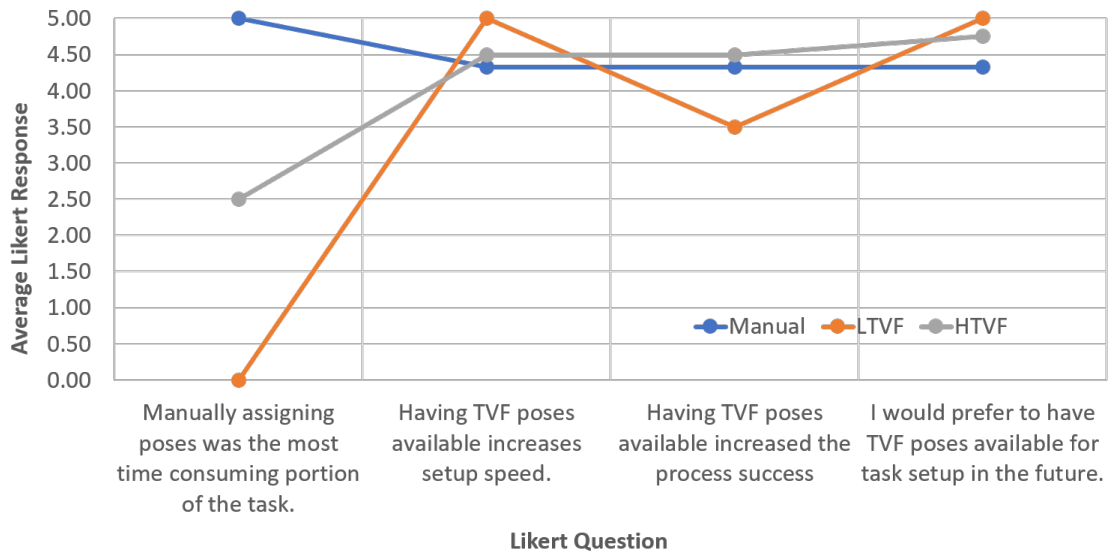


Figure 6.33: Average responses for Likert scale questions five through eight for manual assignment, low resolution TVF, and high resolution TVF testing.

piece and the plasma pen activated. The most significant observation came from User 3 who increased the task execution speed to v10 in order to get the task done more quickly but by decreasing cleaning quality.

6.2.2.7 Summary of Spatially Continuous Task Operator Evaluation

Spatially continuous task operator testing using RobotStudio with a plasma pen and a geometrically complex asymmetric surface led to several conclusion. The first results from limited amount of effort, none completed a manual assignment test while remaining normal to the surface, and operator comments in some trials, such as “I already don’t want to do this anymore.” This task was probably too tedious for engineering professionals with other significant job responsibilities. Also, confidence in the testing conclusions would be much

higher with a larger user pool. Results, such as the average setup time (Figure 6.26), can be very easily skewed with only four operators.

Even with these evaluation shortcomings the results are still highly suggest having TVF poses available for task setup decreases task setup time, operator frustration, and difficulty while increasing task success. Operators were also in agreement they wanted TVF poses available for future task setup after performing a trial where they were available.

6.3 Summary of Task Virtual Fixture Evaluations

This chapter describes three independent experiments to evaluate the consistency, interpretability, and usefulness of TVFs generated with the pipeline developed in Chapter 4 and implemented in Chapter 5. Input data is first checked for inverted $T_{surf}(i).\hat{n}$ regions if applicable (exception is PCN data) and provides operator warnings if necessary (3D Warehouse models and some supertoroids). The next steps in the process transform the incoming data into either a TVF graph structure or RobotStudio compatible file. Analysis of eight superellipsoids, eight superellipsoids, 25 crowdsourced polygonal mesh models, and PCN sensor data showed expected growth rates in the PCN size, intralayer connections, and interlayer connections.

Operator evaluation on spatially discrete and continuous tasks were performed in the MTTT environment and on a hardware system respectively. Results from 11 operators with varying degrees of manipulator experience display increased task reachability during all trials when TVF poses are provided in the first trial. Data also shows the task was less difficult and frustrating for operators when TVF poses are provided in the first trial. The testing pool for hardware evaluation of a spatially continuous task was smaller with four operators.

Quantitative results demonstrate higher plasma clean rates were achieved when TVF poses were available over manually assigning poses. Qualitative data, both Likert surveys and user comments, suggest similar trends with the plasma cleaning task being less difficult and frustrating when TVF poses were available over manually assigning poses. Users also agreed that TVF poses increased setup speed, process success, and were desired for future task setup. Due to the limited test pool and user availability verification of spatially continuous test results will require a larger, more available, test pool with ABB hardware and software training.

Evaluations demonstrate effective TVF generation for superellipsoids, superellipsoids, polygonal meshes, sensor data, and CAD models. Operator testing also shows TVFs are interpretable and assist with task completion. Therefore, TVFs are applicable to complex task geometries including radioactive material, hazardous environment, and emergency response to aid operators with task completion.

Chapter 7

Conclusions and Future Extensions

Decades of nuclear material production prioritization over environmental concerns and nuclear energy production resulted in legacy waste which must be managed, size-reduced, sorted, and properly disposed. The potentially harmful nature of radioactive waste motivates the integration of automation to minimize operator dose [United States Nuclear Regulatory Commission, 2018]. Ideally, robotic system operators will be trained technicians instead of robotics experts despite the effect of the *correspondence problem* and *context switching*. This research effort's goal is to decrease operator burden by advancing VF generation beyond point cloud FRVFs. Constructing layers of point cloud GVFs based on task geometry and execution information around a polygonal mesh FRVF forms a TVF. TVFs provide a more complete representation of the task but allow operators to maintain execution control. GVF layer evaluations with multiple input types and operator evaluation demonstrated TVFs can be generated for complex geometries and are still interpretable to operators. This chapter outlines the algorithms and evaluations presented in previous chapters and discusses research avenues for future development to increase its areas of impact.

7.1 Summary of Task Virtual Fixture Development

Chapter 1 introduces the problem of legacy radioactive waste from multiple origins and automation’s potential to decrease material hazards to operators. It outlines two common barriers to operator adoption of automation systems which are the *correspondence problem* and *context switching*. Advancing semi-autonomous behaviors in the form of VFs generated task geometry and task execution requirements is the proposed solution. This work builds on previous investigations into point cloud FRVFs. The chapter also outlines several closely related topics including teleoperation, graph data structures, ROS, and other associated software packages.

Chapter 2 begins by reviewing robotic efforts utilizing semi-autonomous behaviors to complete nuclear tasks to establish their potential effectiveness in the domain. It also outlines recent efforts in VF research and the similarities to this research thread. The main research parallel is the need to protect critical areas during task execution in both haptic surgical domains and radioactive material tasks involving *high-value* operations. The chapter ends with a detailed analysis of the remaining research and technological gaps like generating constraint geometries effectively.

Chapter 3 discusses the combination of a task defined FRVF and point cloud GVF layers into a singular TVF. TVFs are then demonstrated using volumetric primitives, VNSVFs, and applied to several spatially discrete and spatially continuous tasks. Selecting from precalculated poses with the proper distance and orientation to the task surface reduced operator mental burden during task execution. However, some tasks, either spatially discrete or continuous require geometry more complex than supportable by shape primitives. Thus, this chapter ends outlining the limitations of VNSVFs to fulfill the needs expressed in the

literature.

Chapter 4 outlines the general approach to construct TVFs by combining a polygonal mesh FRVF and layers of point cloud GVFs based on complex task geometry and task execution requirements to meet the needs presented in the literature. Next discussed are TVF input data types and generation algorithms. Calculating and extending surface normals forms point cloud VF layers which become either a FRVF or GVF. Layers of GVFs are converted into the bi-directional TVF graph for storage and operator use.

Chapter 5 implements the algorithms defined in Chapter 4. Input data can either be a binary STL file or PCN sensor data. The TVF generation pipeline, with full intralayer connections and sparse interlayer connections, was written in C++ with the use of third-party libraries including PCL, BGL, OMP, ROS, *RViz*, and *MoveIt!*. The chapter also examines visualization and task execution environments in *RViz* and RobotStudio along with their respective strengths.

Chapter 6 describes TVF evaluations with superellipsoids, superellipsoids, polygonal meshes, sensor data, and CAD models. These tests show the effectiveness of the TVF generation pipeline on all input types. Spatially discrete and spatially continuous operator evaluation studies follows input variations. These studies suggest TVFs are still interpretable to operators and assist significantly with task setup while decreasing task difficulty and user frustration.

7.2 Avenues for Future Development

Outlined in this document is a robust and hardware agnostic TVF generation pipeline based on a polygonal mesh FRVF and layers of point cloud GVFs. Naturally, work remains in the area of semi-autonomous behaviors and VFs. Thus, there are a number of avenues for future integration and development based upon this work.

Dynamic generation - Currently the TVF generation pipeline is run offline, and TVFs are static. However, the pipeline was designed to take PCN sensor data input. Therefore, integration of the TVF generation pipeline into a dynamic VF demonstration with live sensor data will provide capabilities and information unavailable in offline generation and static operation. The capability to generate TVFs based on live sensor data allows *open world* operation when models of the environment may be unavailable *a priori*. Information gathered could include TVF generation times, the effects of variable task parameters, and operator visualization preferences.

Automated path planning - One of the benefits of converting 6 DOF poses into a bi-directional graph are the existing graph algorithms, such as Dijkstra's Shortest Path, available. These algorithms could assist operators with task execution. For example, the operator could select all of an inspection task's relevant poses and leave the system to calculate the shortest path to gather data.

Contact tasks - The tasks investigated in this research effort were non-contact tasks. However, there are possibilities to extend TVF applicability to contact tasks through previously developed force control frameworks. This pairing could reduce the operator's mental burden for tasks such as surface contamination swabbing, surface treatments, and

part assembly.

Machine learning - TVF poses could provide an underlying framework for various types of machine learning algorithms. Recordings of operator task execution could provide the relevant TVF poses for task execution. The generated task path could then be adjusted by the operator based on their preferences and used to update future path selection.

Surface region subgraphs - The TVF generation pipeline considers the surface as a whole during normal extension, interpolation, and voxelization. One possible method for reducing distortions in GVF layers is to segment the task surface into regions and apply layer generation algorithms independently. The simplest example of this approach is a cube where is side is treated independently before being converted into subgraphs of the TVF graph. This approach could increase interpolation and voxelization effectiveness while also reducing intralayer connections by only connecting regions to a single vertex in neighboring regions. Additionally, there might be operator visualization benefits by reducing the number of visualized points but maintaining the ability to navigate a task surface quickly.

Surface reconstruction algorithms - Currently the TVF generation pipeline minimizes surface reconstruction, interpolation, and voxelization to reduce distortions in layers which are likely non-manifold. The addition of reconstruction algorithms, such as point cloud interpolation accounting for curvature using spatially aware neighborhoods, could improve the output graph. Such TVF generation upgrades would likely be most successful at improving output graphs when paired with surface region subgraphs.

Extended workspace and path calculations - The MTTT interface calculates IK and planning scene reachability which is integrated into TVF pose visualization. Additional

visualizations of manipulator singularities and path reconfigurations would most likely increase operator comfort with task execution.

7.3 Summary and Significance of Proposed Research

This document presents VF generation advancements to increase semi-autonomous behavior assistance and decrease operator mental burden. The major contributions of this research are:

- **The TVF generation pipeline** - A general, task information based, VF generator for complex geometry. The software pipeline will be released open source once approved by LANL.
- **The Manipulator to Task Transform Tool *RViz* interface** - A visualization and TVF navigation interface for task execution integrating ROS, *RViz*, *MoveIt!*, and reachability analysis.
- **ABB RobotStudio Integration** - TVF information was integrated with ABB's proprietary RobotStudio to demonstrate hardware and ROS agnosticism.
- **Multiple Input Data Sources** - The TVF generation pipeline was tested with superellipsoids, superellipsoids, polygonal meshes, sensor data, and CAD models which demonstrated expected results.
- **Spatially Discrete Operator Evaluation** - Operator evaluations with the MTTT interface conclude TVFs generated from complex geometry are still operator interpretable and decrease task difficulty and frustration.
- **Spatially Continuous Operator Evaluation** - Operator evaluations with the RobotStudio interface conclude TVF poses decrease task setup time, difficulty, and frustration.

These contributions provide hardware agnostic TVFs from complex geometries for operator assistance with task execution. While tasks are not fully automated, a foundation is provided for avenues of future research to increase VF assistance further. The TVF software pipeline and MTTT interface will be released open source through Github for use in robotics research and community improvement once approved by LANL.

Appendix A

Mobile Manipulator Base Location Variable Normal Surface Virtual Fixture Heat Maps

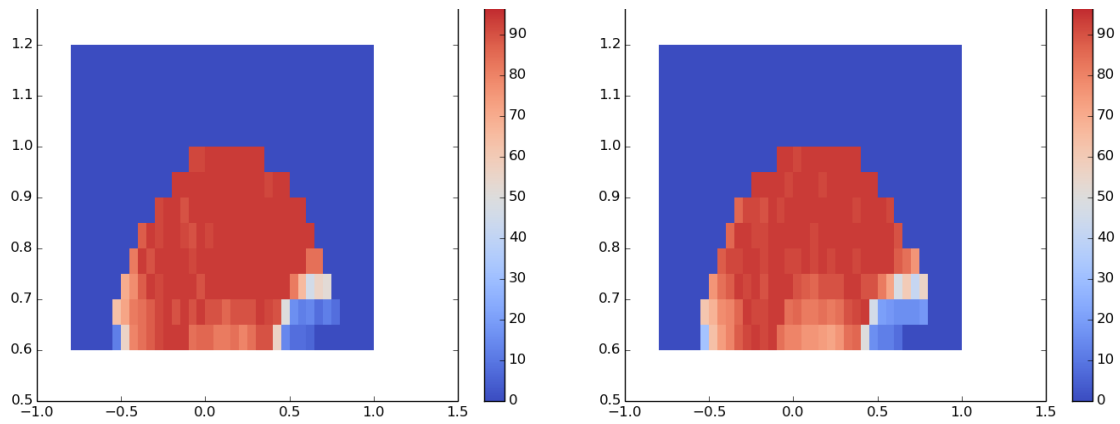


Figure A.1: Mobile base placement for task execution results at 20 cm (left) and 25 cm (right) displayed as a heat map scaled in meters.

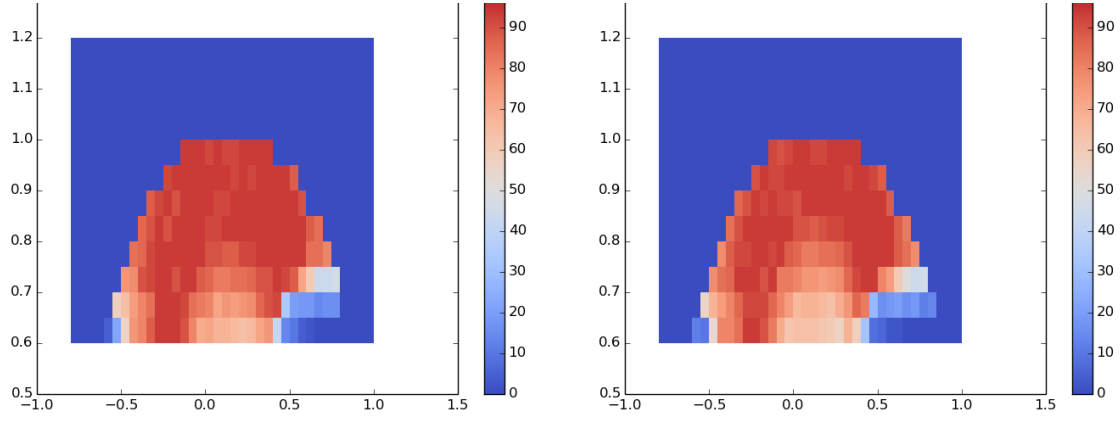


Figure A.2: Mobile base placement for task execution results at 30 cm (left) and 35 cm (right) displayed as a heat map scaled in meters.

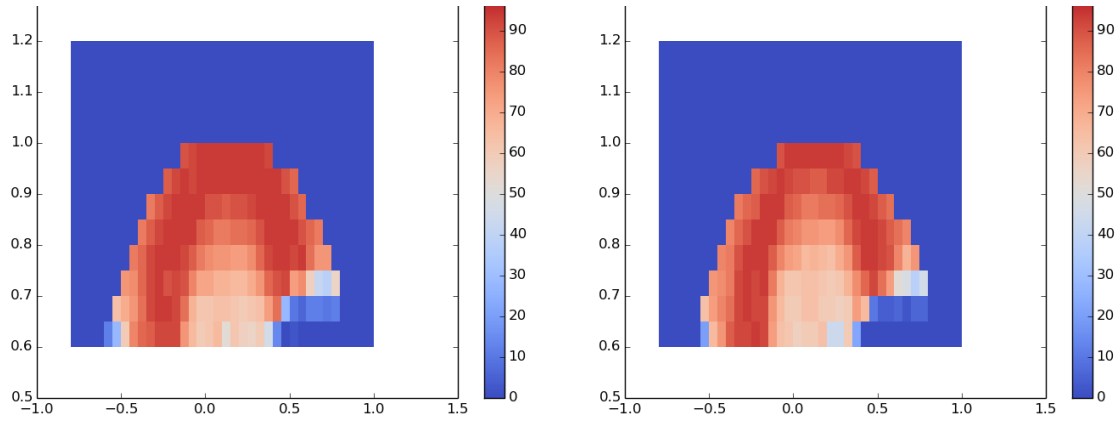


Figure A.3: Mobile base placement for task execution results at 40 cm (left) and 45 cm (right) displayed as a heat map scaled in meters.

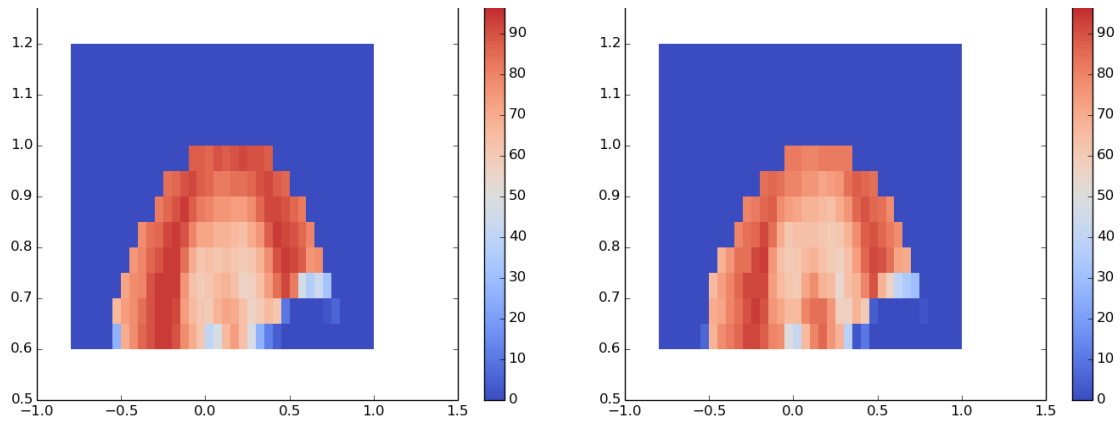


Figure A.4: Mobile base placement for task execution results at 50 cm (left) and 55 cm (right) displayed as a heat map scaled in meters.

Appendix B

Surface Normals with Varying k Nearest Neighbor Values

Table B.1: Normal Angles for Points in Figure 4.9 using Least Square Method (Equation 4.4) with $k = 3$

Point	x	y	\bar{X}	\bar{Y}	$\Sigma(x - \bar{X})(y - \bar{Y})$	$\Sigma(x - \bar{X})^2$	Θ	Normal
1	0.00	0.00	0.25	0.00	0.00	0.69	0.00	1.57
2	0.50	0.00	0.50	0.00	0.00	1.50	0.00	1.57
3	1.00	0.00	1.00	0.00	0.00	2.50	0.00	1.57
4	1.50	0.00	1.50	0.00	0.00	1.75	0.00	1.57
5	2.00	0.00	1.83	0.17	0.08	0.17	0.46	2.03
6	2.00	0.50	2.17	0.33	0.08	0.17	0.46	2.03
7	2.50	0.50	2.50	0.50	0.00	0.50	0.00	1.57
8	3.00	0.50	2.83	0.33	-0.08	0.17	-0.46	1.11
9	3.00	0.00	3.17	0.17	-0.08	0.17	-0.46	1.11
10	3.50	0.00	3.50	0.00	0.00	0.50	0.00	1.57
11	4.00	0.00	4.00	0.00	0.00	0.50	0.00	1.57
12	4.50	0.00	4.50	0.00	0.00	0.50	0.00	1.57
13	5.00	0.00	4.83	0.17	0.08	0.17	0.46	2.03
14	5.00	0.50	5.17	0.33	0.08	0.17	0.46	2.03
15	5.50	0.50	5.50	0.50	0.00	0.50	0.00	1.57
16	6.00	0.50	5.83	0.33	-0.08	0.17	-0.46	1.11
17	6.00	0.00	6.17	0.17	-0.08	0.17	-0.46	1.11
18	6.50	0.00	6.50	0.00	0.00	0.50	0.00	1.57
19	7.00	0.00	7.00	0.00	0.00	0.50	0.00	1.57
20	7.50	0.00	7.50	0.00	0.00	0.50	0.00	1.57
21	8.00	0.00	7.75	0.00	0.00	0.13	0.00	1.57

Table B.2: Normal Angles for Points in Figure 4.9 using Least Square Method (Equation 4.4) with $k = 5$

Point	x	y	\bar{X}	\bar{Y}	$\Sigma(x - \bar{X})(y - \bar{Y})$	$\Sigma(x - \bar{X})^2$	Θ	Normal
1	0.00	0.00	0.50	0.00	0.00	0.50	0.00	1.57
2	0.50	0.00	0.75	0.00	0.00	1.25	0.00	1.57
3	1.00	0.00	1.00	0.00	0.00	2.50	0.00	1.57
4	1.50	0.00	1.40	0.10	0.30	1.70	0.17	1.75
5	2.00	0.00	1.80	0.20	0.45	1.30	0.33	1.90
6	2.00	0.50	2.20	0.30	0.45	1.30	0.33	1.90
7	2.50	0.50	2.50	0.30	0.00	1.00	0.00	1.57
8	3.00	0.50	2.80	0.30	-0.45	1.30	-0.33	1.24
9	3.00	0.00	3.20	0.20	-0.45	1.30	-0.33	1.24
10	3.50	0.00	3.60	0.10	-0.30	1.70	-0.17	1.40
11	4.00	0.00	4.00	0.00	0.00	2.50	0.00	1.57
12	4.50	0.00	4.40	0.10	0.30	1.70	0.17	1.75
13	5.00	0.00	4.80	0.20	0.45	1.30	0.33	1.90
14	5.00	0.50	5.20	0.30	0.45	1.30	0.33	1.90
15	5.50	0.50	5.50	0.30	0.00	1.00	0.00	1.57
16	6.00	0.50	5.80	0.30	-0.45	1.30	-0.33	1.24
17	6.00	0.00	6.20	0.20	-0.45	1.30	-0.33	1.24
18	6.50	0.00	6.60	0.10	-0.30	1.70	-0.17	1.40
19	7.00	0.00	7.00	0.00	0.00	2.50	0.00	1.57
20	7.50	0.00	7.25	0.00	0.00	53.81	0.00	1.57
21	8.00	0.00	7.50	0.00	0.00	56.75	0.00	1.57

Table B.3: Normal Angles for Points in Figure 4.9 using Least Square Method (Equation 4.4) with $k = 7$

Point	x	y	\bar{X}	\bar{Y}	$\Sigma(x - \bar{X})(y - \bar{Y})$	$\Sigma(x - \bar{X})^2$	Θ	Normal
1	0.00	0.00	0.75	0.00	0.00	1.25	0.00	1.57
2	0.50	0.00	1.00	0.00	0.00	2.50	0.00	1.57
3	1.00	0.00	1.17	0.08	0.42	3.33	0.12	1.70
4	1.50	0.00	1.36	0.14	0.89	4.86	0.18	1.75
5	2.00	0.00	1.79	0.21	1.07	4.43	0.24	1.81
6	2.00	0.50	2.14	0.21	0.54	3.36	0.16	1.73
7	2.50	0.50	2.50	0.21	0.00	3.00	0.00	1.57
8	3.00	0.50	2.86	0.21	-0.54	3.36	-0.16	1.41
9	3.00	0.00	3.21	0.21	-1.07	4.43	-0.24	1.33
10	3.50	0.00	3.64	0.14	-0.89	4.86	-0.18	1.39
11	4.00	0.00	4.00	0.14	0.00	4.50	0.00	1.57
12	4.50	0.00	4.36	0.14	0.89	4.86	0.18	1.75
13	5.00	0.00	4.79	0.21	1.07	4.43	0.24	1.81
14	5.00	0.50	5.14	0.21	0.54	3.36	0.16	1.73
15	5.50	0.50	5.50	0.21	0.00	3.00	0.00	1.57
16	6.00	0.50	5.86	0.21	-0.54	3.36	-0.16	1.41
17	6.00	0.00	6.21	0.21	-1.07	4.43	-0.24	1.33
18	6.50	0.00	6.64	0.14	-0.89	4.86	-0.18	1.39
19	7.00	0.00	6.83	0.08	-0.42	3.33	-0.12	1.45
20	7.50	0.00	7.00	0.00	0.00	2.50	0.00	1.57
21	8.00	0.00	7.25	0.00	0.00	1.25	0.00	1.57

Table B.4: Normal Angles for Points in Figure 4.9 using Least Square Method (Equation 4.4) with $k = 11$

Point	x	y	\bar{X}	\bar{Y}	$\Sigma(x - \bar{X})(y - \bar{Y})$	$\Sigma(x - \bar{X})^2$	Θ	Normal
1	0.00	0.00	1.17	0.08	0.42	3.33	0.12	1.70
2	0.50	0.00	1.36	0.14	0.89	4.86	0.18	1.75
3	1.00	0.00	1.56	0.19	1.41	7.22	0.19	1.76
4	1.50	0.00	1.72	0.17	1.17	9.06	0.13	1.70
5	2.00	0.00	1.90	0.15	0.90	11.90	0.08	1.65
6	2.00	0.50	2.09	0.14	0.61	15.91	0.04	1.61
7	2.50	0.50	2.50	0.14	0.00	15.50	0.00	1.57
8	3.00	0.50	2.91	0.14	-0.61	15.91	-0.04	1.53
9	3.00	0.00	3.27	0.18	-0.30	15.18	-0.02	1.55
10	3.50	0.00	3.64	0.23	-0.09	15.55	-0.01	1.56
11	4.00	0.00	4.00	0.27	0.00	17.00	0.00	1.57
12	4.50	0.00	4.36	0.23	0.09	15.55	0.01	1.58
13	5.00	0.00	4.73	0.18	0.30	15.18	0.02	1.59
14	5.00	0.50	5.09	0.14	0.61	15.91	0.04	1.61
15	5.50	0.50	5.50	0.14	0.00	15.50	0.00	1.57
16	6.00	0.50	5.91	0.14	-0.61	15.91	-0.04	1.53
17	6.00	0.00	6.10	0.15	-0.90	11.90	-0.08	1.50
18	6.50	0.00	6.28	0.17	-1.17	9.06	-0.13	1.44
19	7.00	0.00	6.44	0.19	-1.41	7.22	-0.19	1.38
20	7.50	0.00	6.64	0.14	-0.89	4.86	-0.18	1.39
21	8.00	0.00	6.83	0.08	-0.42	3.33	-0.12	1.45

Appendix C

Crowd Sourced Polygonal Meshes

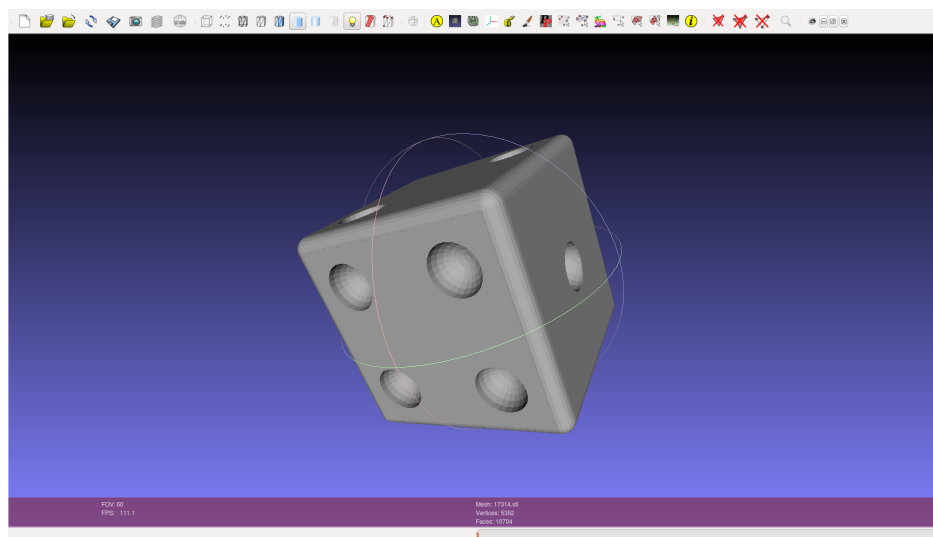


Figure C.1: Thingiverse model 17314, <https://www.thingiverse.com/thing:17314>

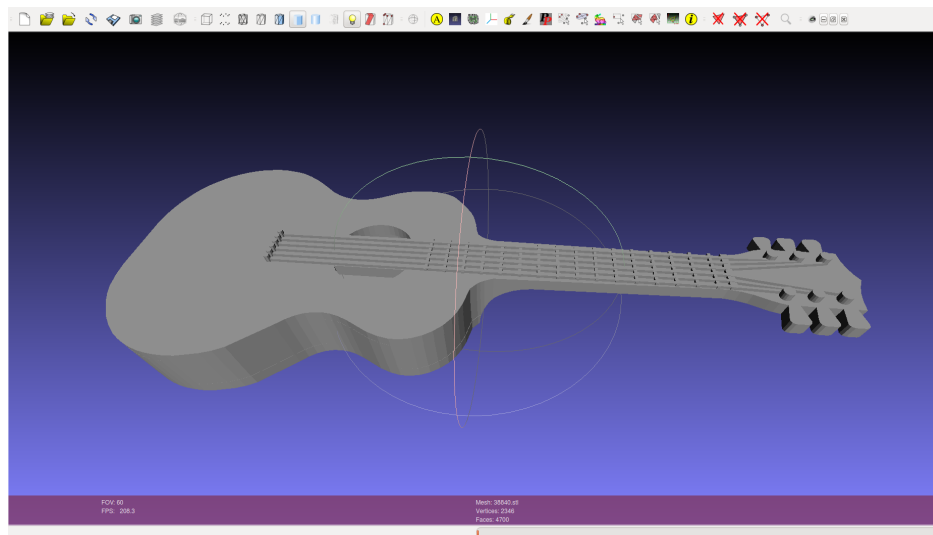


Figure C.2: Thingiverse model 38840, <https://www.thingiverse.com/thing:38840>

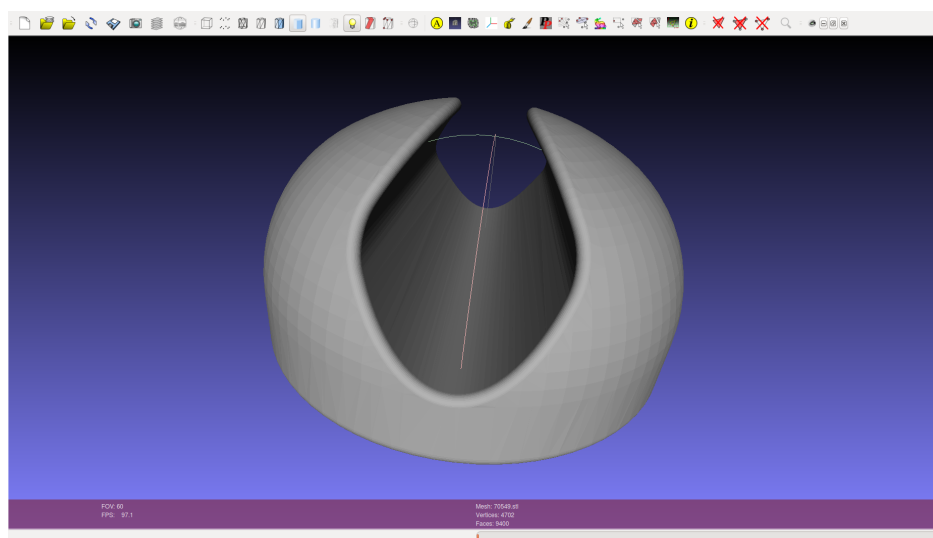


Figure C.3: Thingiverse model 70549, <https://www.thingiverse.com/thing:70549>

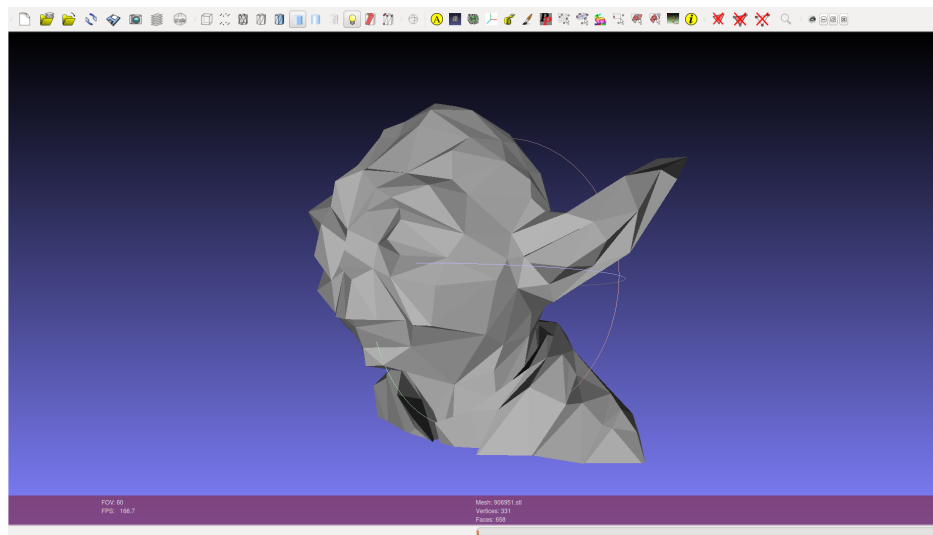


Figure C.4: Thingiverse model 906951, <https://www.thingiverse.com/thing:906951>

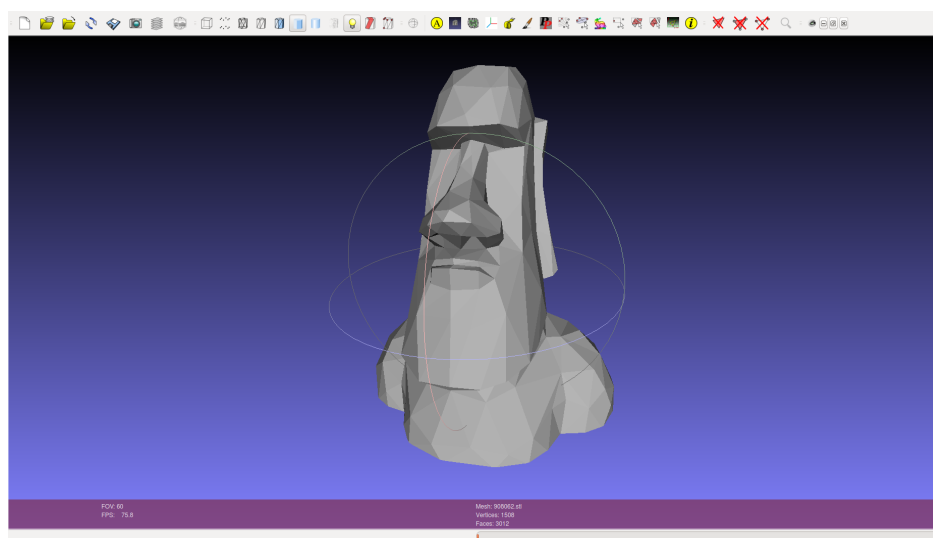


Figure C.5: Thingiverse model 908062, <https://www.thingiverse.com/thing:908062>

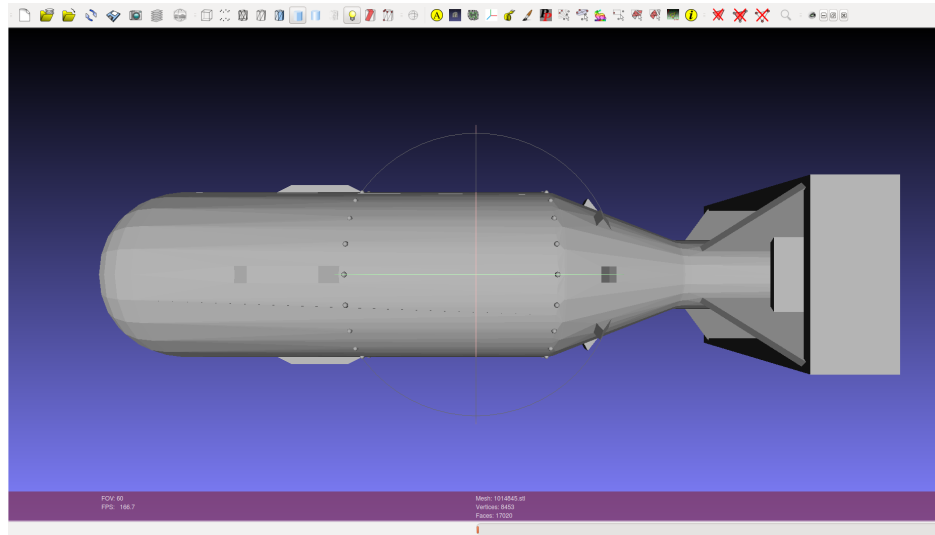


Figure C.6: Thingiverse model 1014845, <https://www.thingiverse.com/thing:1014845>

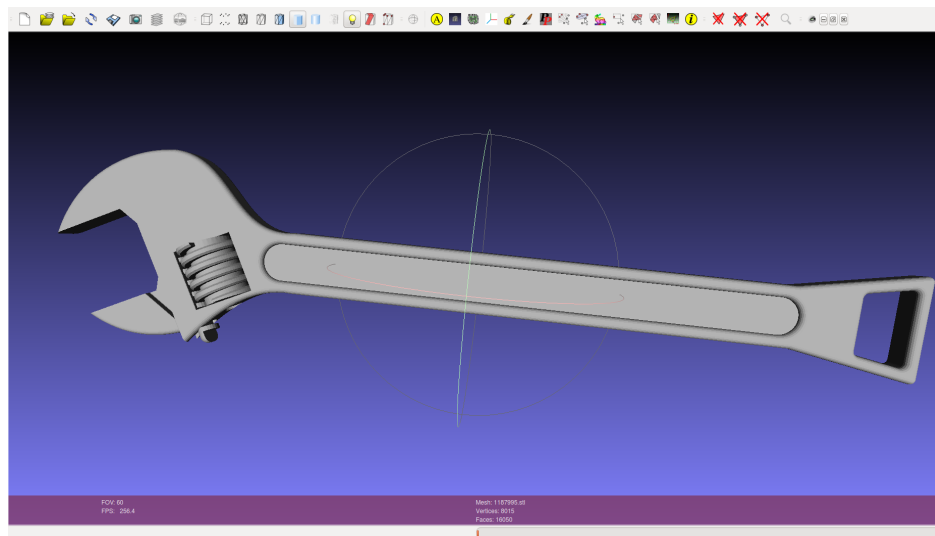


Figure C.7: Thingiverse model 1187995, <https://www.thingiverse.com/thing:1187995>

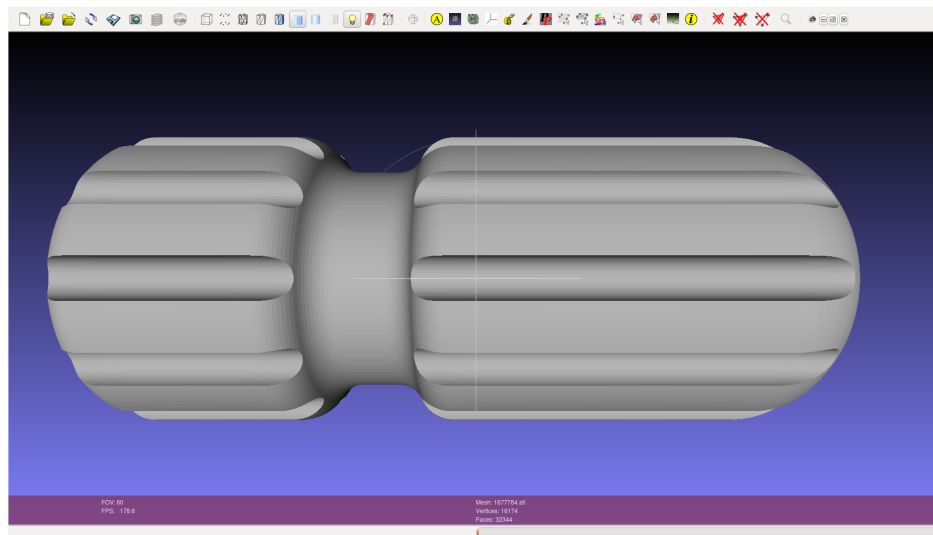


Figure C.8: Thingiverse model 1677784, <https://www.thingiverse.com/thing:1677784>

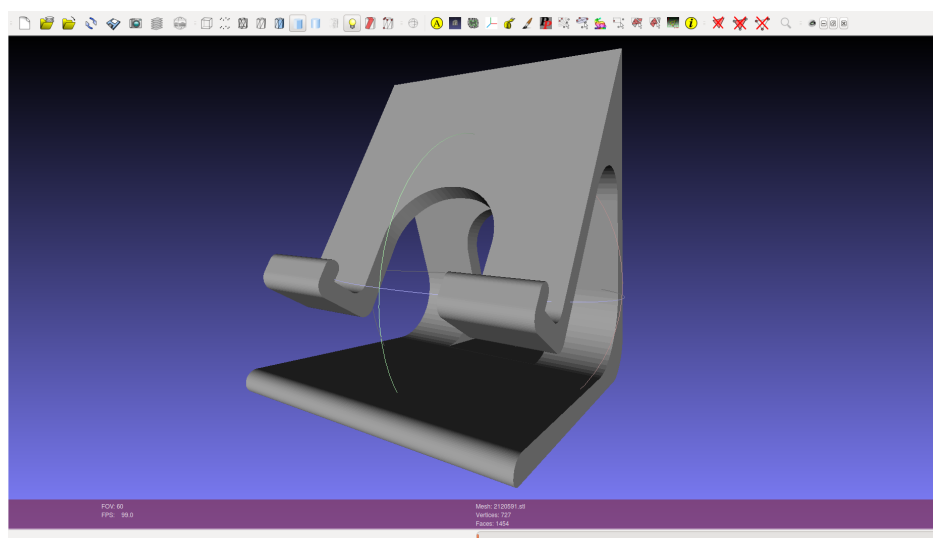


Figure C.9: Thingiverse model 2120591, <https://www.thingiverse.com/thing:2120591>

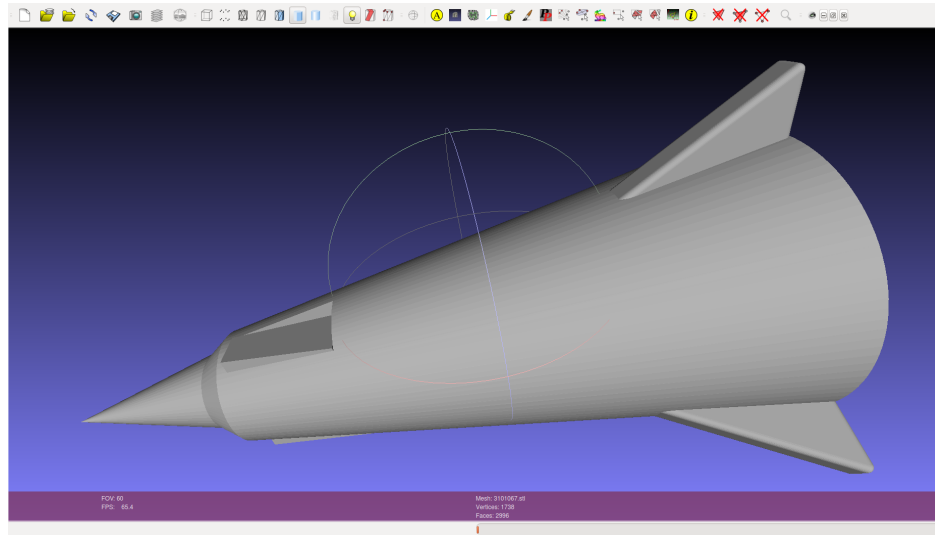


Figure C.10: Thingiverse model 3101067, <https://www.thingiverse.com/thing:3101067>

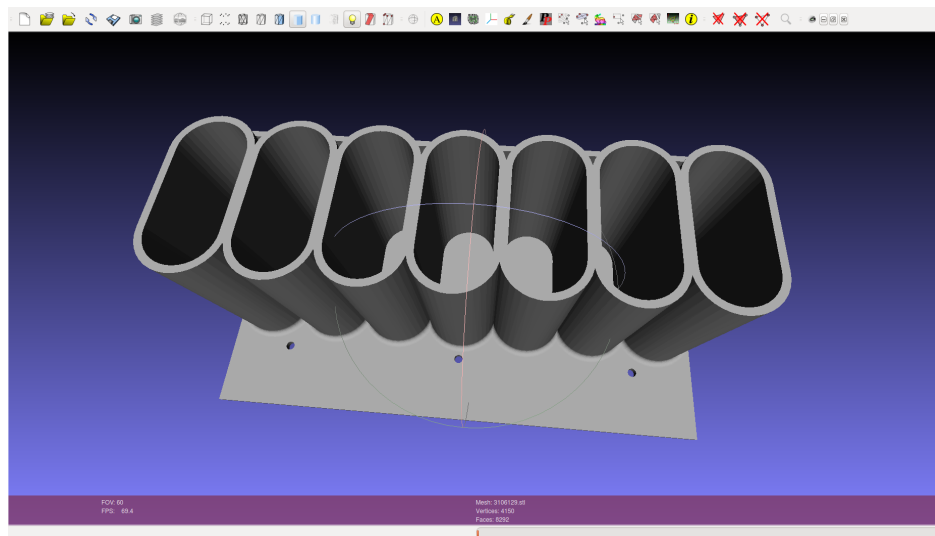


Figure C.11: Thingiverse model 3106129, <https://www.thingiverse.com/thing:3106129>

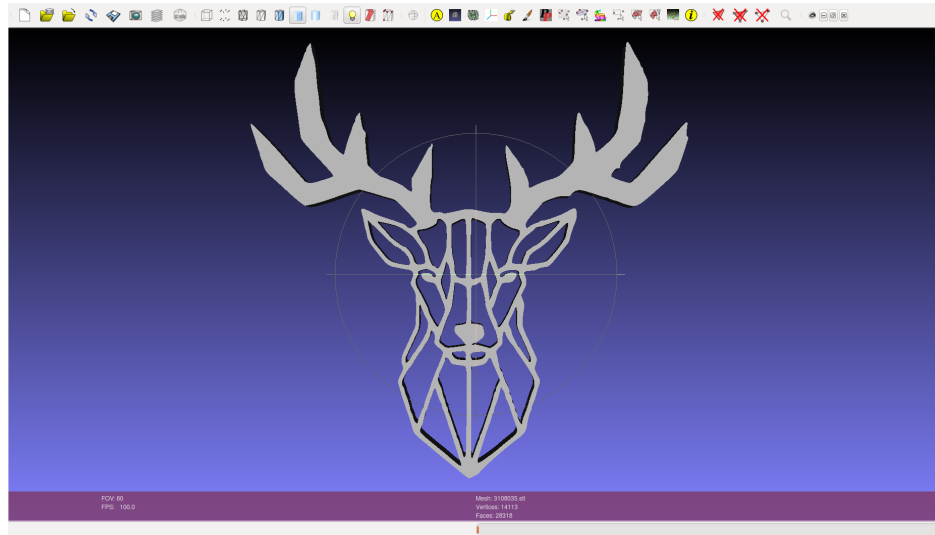


Figure C.12: Thingiverse model 3108035, <https://www.thingiverse.com/thing:3108035>

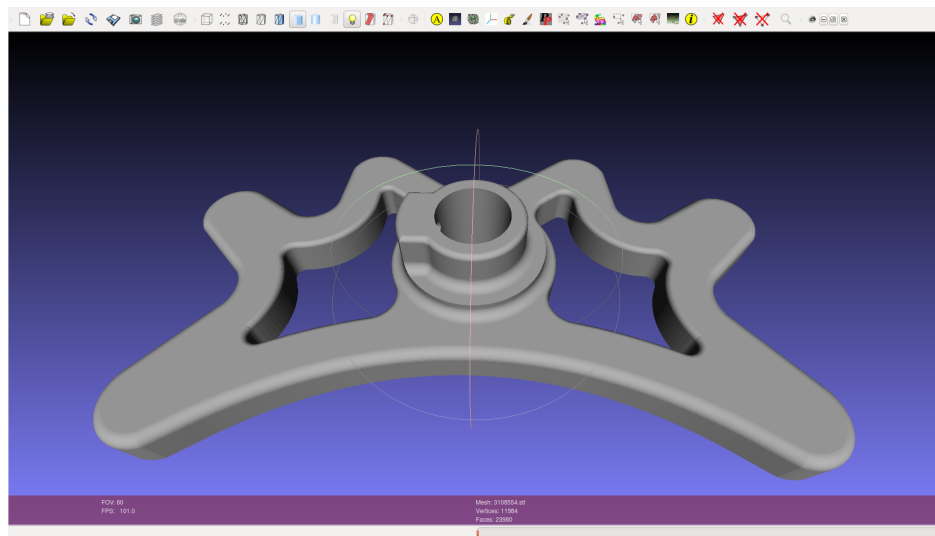


Figure C.13: Thingiverse model 3108554, <https://www.thingiverse.com/thing:3108554>

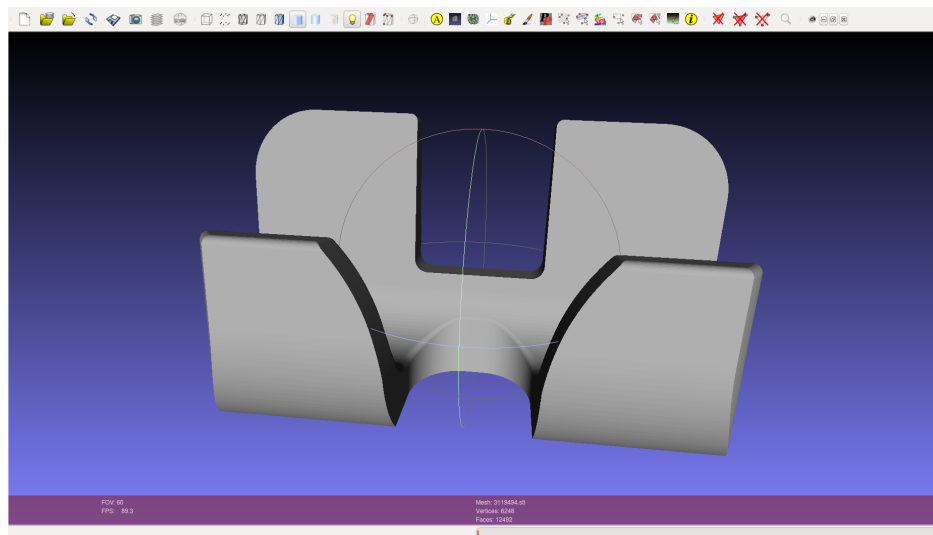


Figure C.14: Thingiverse model 3119494, <https://www.thingiverse.com/thing:3119494>

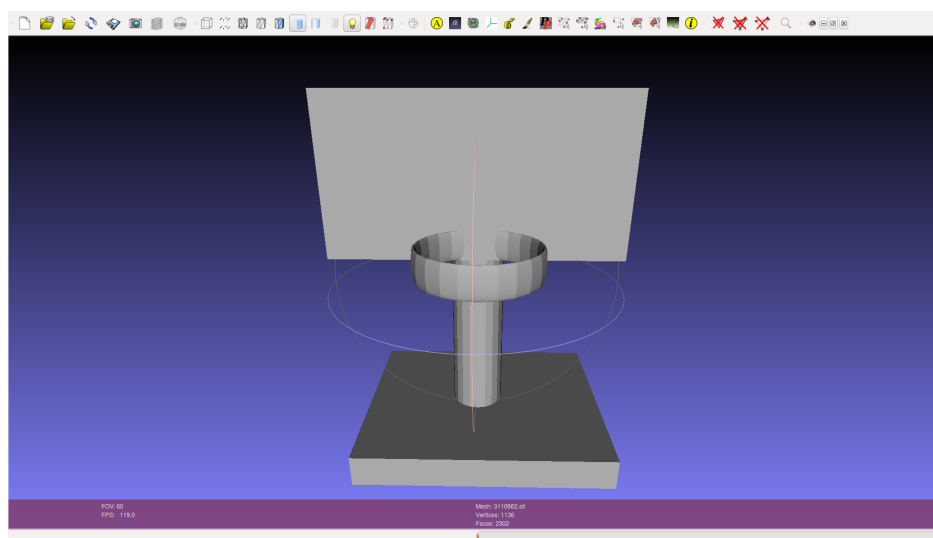


Figure C.15: Thingiverse model 3110862, <https://www.thingiverse.com/thing:3110862>

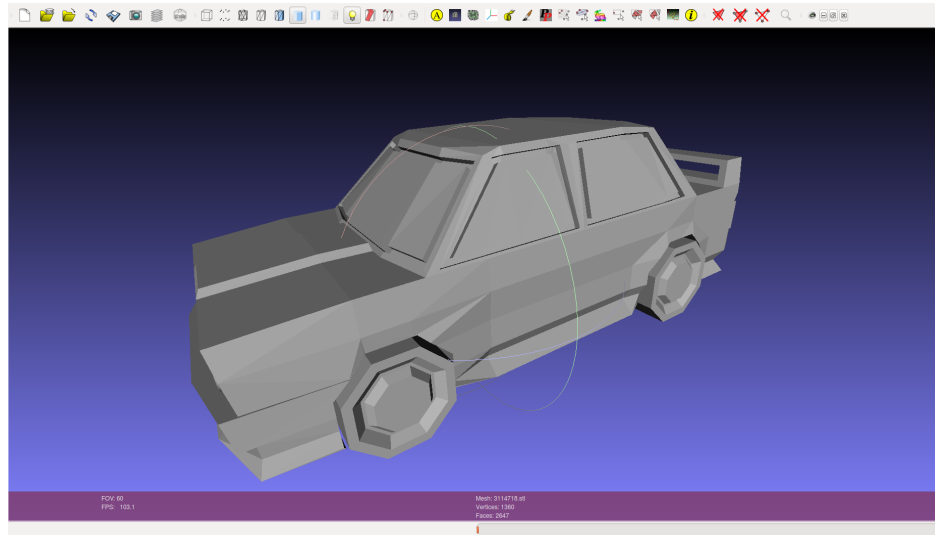


Figure C.16: Thingiverse model 3114718, <https://www.thingiverse.com/thing:3114718>

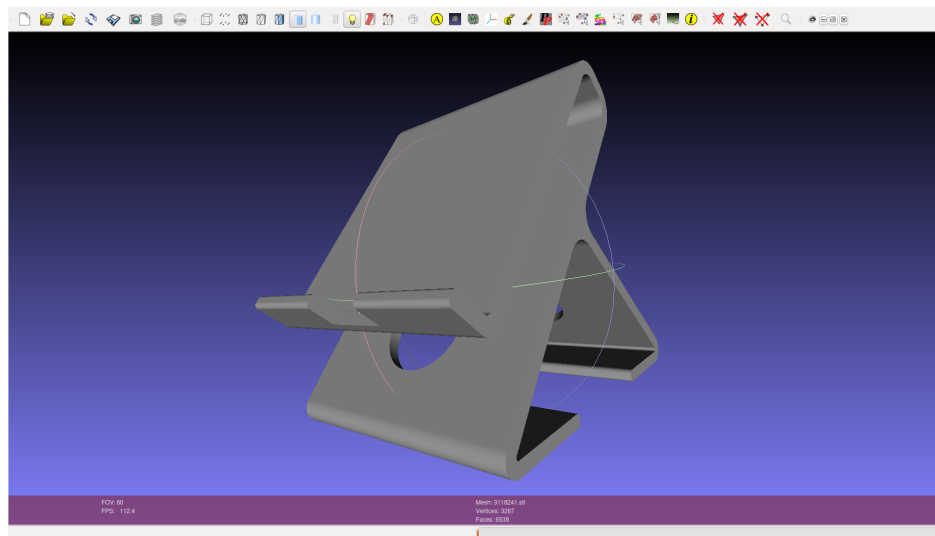


Figure C.17: Thingiverse model 3118241, <https://www.thingiverse.com/thing:3118241>

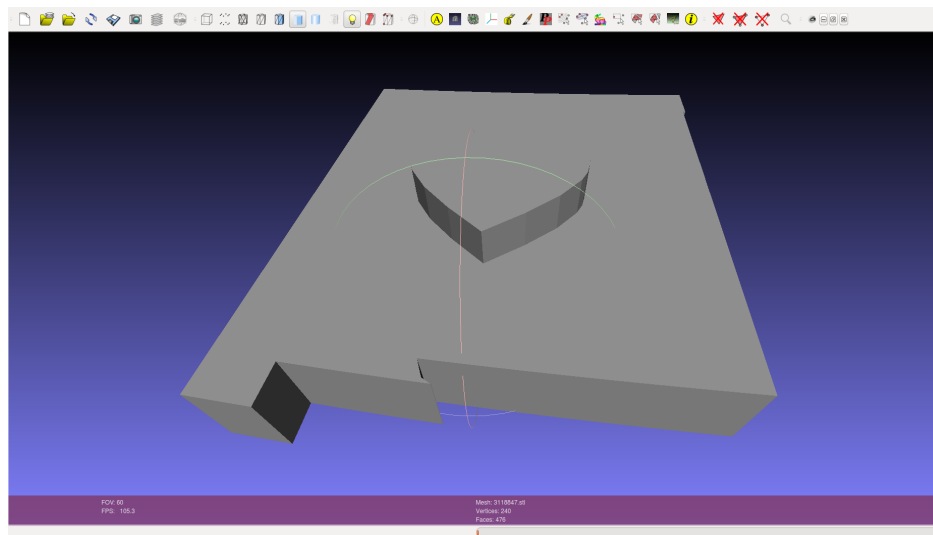


Figure C.18: Thingiverse model 3118847, <https://www.thingiverse.com/thing:3118847>

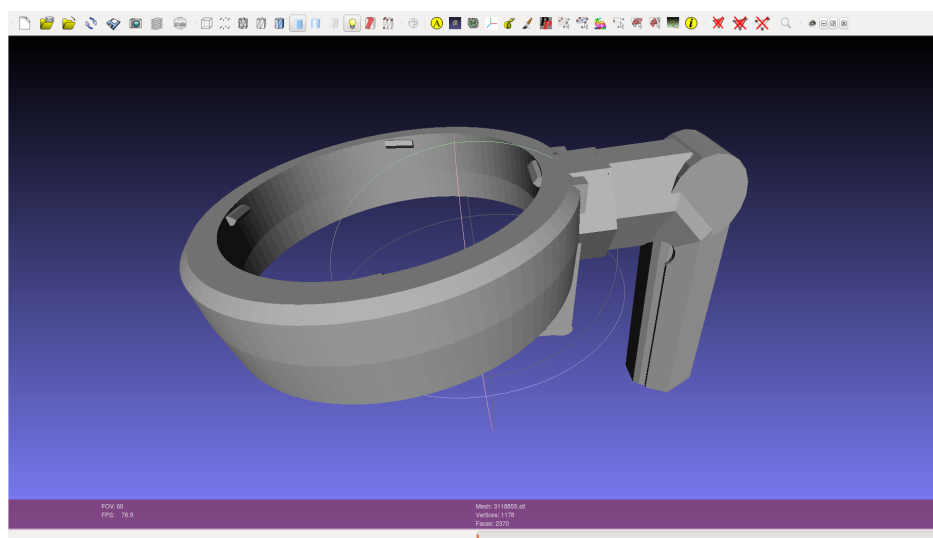


Figure C.19: Thingiverse model 3118855, <https://www.thingiverse.com/thing:3118855>

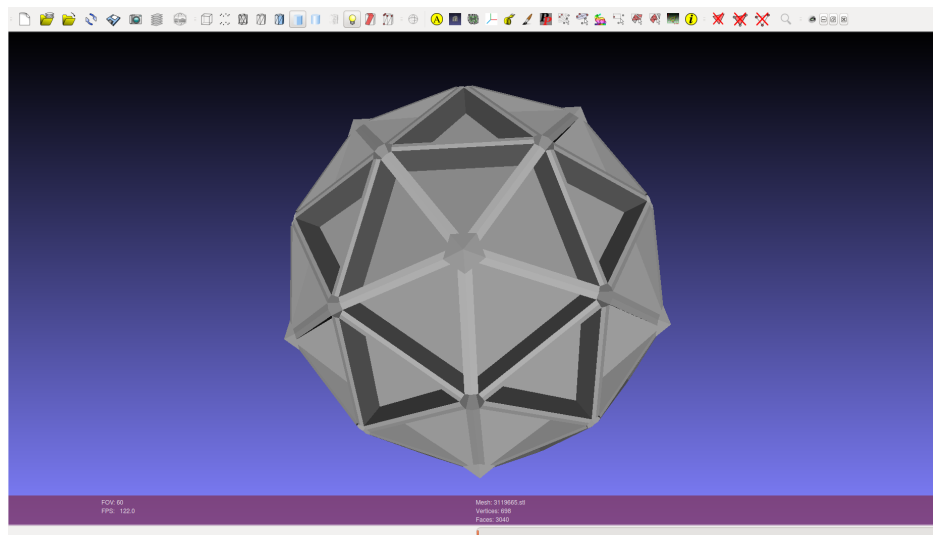


Figure C.20: Thingiverse model 3119665, <https://www.thingiverse.com/thing:3119665>

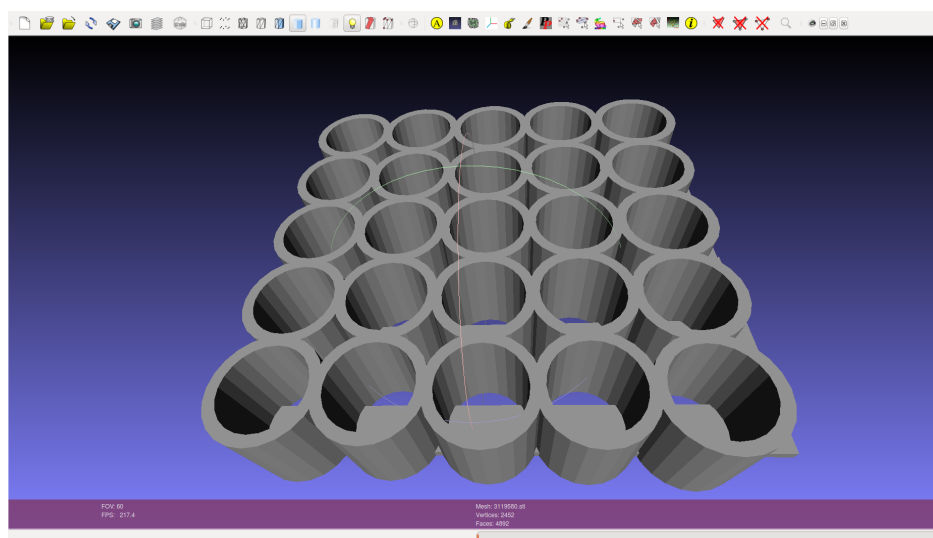


Figure C.21: Thingiverse model 3119580, <https://www.thingiverse.com/thing:3119580>

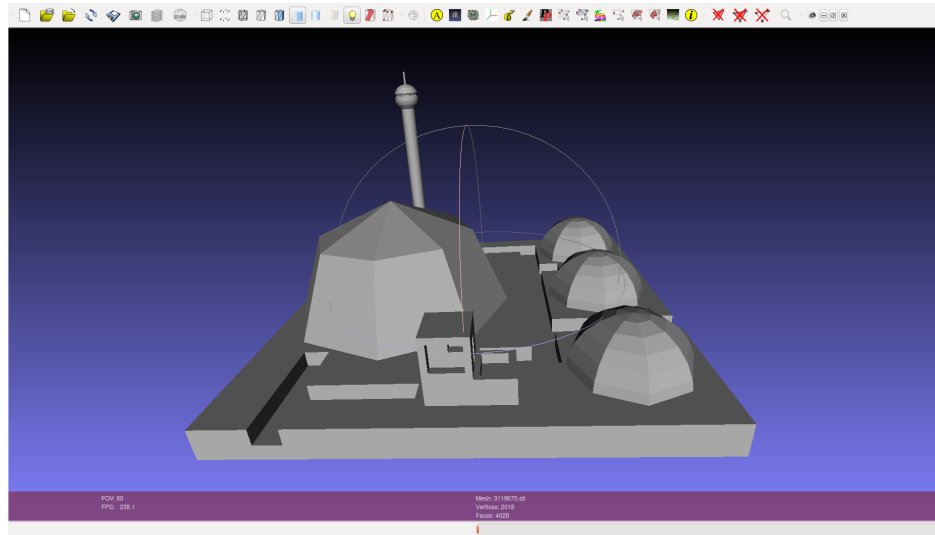


Figure C.22: Thingiverse model 3119670, <https://www.thingiverse.com/thing:3119670>

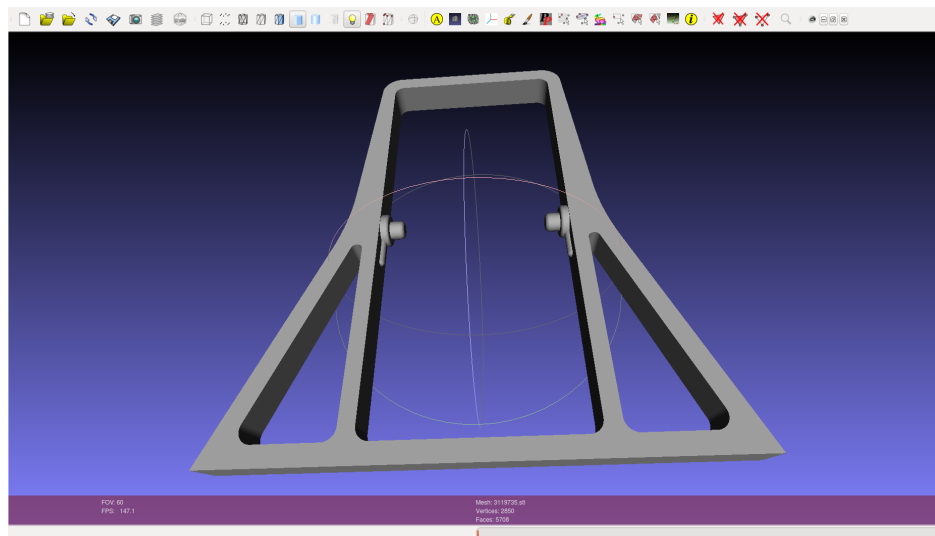


Figure C.23: Thingiverse model 3119735, <https://www.thingiverse.com/thing:3119735>

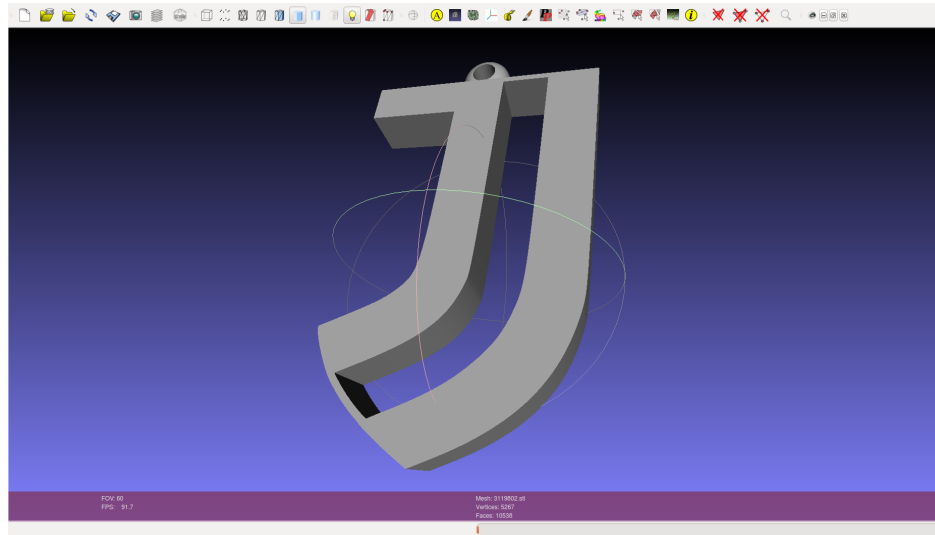


Figure C.24: Thingiverse model 3119802, <https://www.thingiverse.com/thing:3119802>

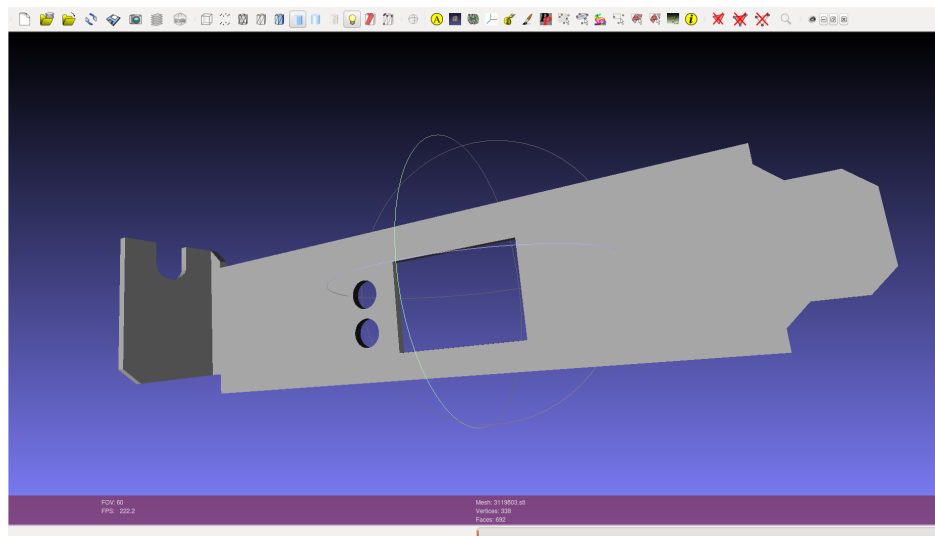


Figure C.25: Thingiverse model 3119803, <https://www.thingiverse.com/thing:3119803>

Appendix D

Institutional Review Board Documents

Research Proposal

1. Title
Verification of Task Virtual Fixtures Through Operator Evaluation
2. Principal Investigator
Andrew Sharp, aps2325, Mechanical Engineering
3. Purpose
A new method for task virtual fixture (TVF) generation from CAD has been developed. Virtual fixtures are virtual constraints on robotic motion much like a physical jig. The first test will compare field experts using standard programming interfaces to ascertain the benefits of TVFs. Four trail methods have been chosen to verify the expected increase in task setup efficiency with TVFs. The trail methods include manual setup, integrating a commercial force control package, TVFs, and TVFs integrated with force control. The second test examines increases in the efficiency of task model placement through a tool build upon TVFs where the goal is to have the greatest task portion within the robot's reach. A group non-experts will be added to the testing pool for this portion of testing. TVF assistance will be compared to current iterative placement methods. The hypothesis suggests higher levels of operator assistance will speed up task set up.
4. Procedures
For the first test expert users with a significant amount of industrial manipulator experience will be provided with a surface to clean with a non-contact plasma pen attached to a 6 degree of freedom manipulator. During this portion of testing the robot and plasma pen will be operating but will be surrounded by a safety screen and all users will be located several feet out of the robot's workspace. Due to the small size of the available expert user group half will test the two control methods first and half will test the two TVF assisted methods first. The four methods are:
 - Manual setup of the robot poses
 - Manual setup of the robot poses with a contact finger and integrated force control
 - TVF poses will be available to the operator
 - TVF poses will be available to the operator with a contact finger and integrated force control

Participants will be asked to place a series of 3D models in the robotic manipulators workspace for the second test. Model placement will take place in a virtual environment presenting and use computer peripherals, presenting no potential danger to the subjects. The goal of the model placement will be for the manipulator reach the highest percentage of the task surface or task path. Testing will be expanded to include a group of non-expert individuals with little industrial manipulator experience. Each experience level will be divided into two groups with one group placing the model without reachability feedback and the other being provided with live reachability feedback.

 - a. Location
Data collection will occur in room 15 of the SM-39 building on the Los Alamos National Laboratory research campus.
 - b. Resources

Figure D.1: IRB proposal for TVF operator evaluation page one.

Research Proposal

No funds/resources are required to gather the required data. The PI's research is supported by the Los Alamos National Laboratory.

- c. Study Timeline
 - One semester

5. Measures

The measurements collected for the method comparison test will consist of several time measurements (setup time, run time), success of the task (surface percentage cleaned), number of collisions or excessive forces, a questionnaire of Likert scale questions after each trial, another questionnaire at the end of the test (see attached). During testing participants will also be able to provide their comments on the process to the PI. These comments will only be recorded by expert or non-expert level.

Several measurements will be used to evaluate the success of the model placement. These include the time for model placement before evaluation. Evaluation will consist of calculations for either the reachable percentage of the TVF or the task path. Following the tests there will also be a survey of Likert scale questions designed for this experiment (see attached). During testing participants will also be able to provide their comments on the process to the PI. These comments will only be recorded by expert or non-expert level.

6. Participants

- a. Target Population
 - Staff and student interns at Los Alamos National Laboratory
- b. Inclusion/Exclusion
 - None
- c. Benefits
 - Simplify setup for future robotic tasks and provide more intuitive interfaces.
- d. Risks
 - None
- e. Recruitment
 - Informal discussions with co-workers at Los Alamos National Lab.
- f. Obtaining Informed Consent
 - Consent forms will be used (see attachment). The PI will be present to answer any questions participants may have.

7. Privacy and Confidentiality

No personally identifying data will be collected. Additionally, data will be kept on password protected media.

8. Compensation

None

Figure D.2: IRB proposal for TVF operator evaluation page two.



memorandum

Pit Technologies Division, Assembly Group

To: Dr. James Wilson, Ph.D., IRB Chair
University of Texas at Austin, Texas 78713
irbchair@austin.utexas.edu

From: David Gubernatis, PT-3, E513 *DL*

Phone: 505-665-0925

Symbol: PT-3: 18-0011

Date: August 25, 2018

Subject: Andrew Sharp-Permission Granted to conduct Research at LANL

Dear Dr. Wilson:

The purpose of this letter is to grant Andrew Sharp, a graduate research assistant at the University of Texas at Austin permission to conduct research at Los Alamos National Laboratory.

The project, "Verification of Task Virtual Fixtures through Operator Evaluation" entails operator evaluation of virtual fixtures generated from CAD data. Two user groups containing approximately fifteen test subjects will take part in two tests to determine the usefulness of a newly developed pipeline for generating virtual fixtures. The first test will compare field experts using standard programming interfaces to ascertain the benefits of these virtual fixtures. The second test, participants will expand to include non-experts.

Their goal will be to place a task model in a virtual robot workspace in position enabling task completion. These tests will be performed at Los Alamos National Laboratory at SM-39. PT-3 was selected because of their ongoing relationship with the Nuclear Robotics Group at UT Austin.

Testing results will be provided and reviewed by Los Alamos National Laboratory prior to publication.

Andrew Sharp has had a GRA position during past summers and started working full time with PT-3 in January.

I, Dave Gubernatis do hereby grant permission for Andrew Sharp to conduct Verification of Task Virtual Fixtures through Operator Evaluation at PT-3 facilities, SM-39.

DCG:vm

Copy: PT-3 Correspondence file

An Equal Opportunity Employer / Operated by Los Alamos National Security, LLC for the U.S. Department of Energy's NNSA



Figure D.3: IRB site approval letter for TVF operator evaluation.

IRB USE ONLY
Study Number: 2018-06-0092
Approval Date: 08/27/2018
Name of Funding Agency (if applicable): Los Alamos National Laboratory

Consent for Participation in Research

Title: Verification of Task Virtual Fixtures Through Operator Evaluation

Introduction

The purpose of this form is to provide you information that may affect your decision as to whether or not to participate in this research study. The person performing the research will answer any of your questions. Read the information below and ask any questions you might have before deciding whether or not to take part. If you decide to be involved in this study, this form will be used to record your consent.

Purpose of the Study

You have been asked to participate in a research study about virtual fixtures for shared robot control. The purpose of this study is to determine the effectiveness of newly created tools for accelerating task set up in a robot's workspace compared to current methods.

What will you be asked to do?

If you agree to participate in this study, you will be asked to either set up a surface cleaning task using four different methods or place part models in a virtual robot's workspace via computer peripherals. The study will take several hours to set up surface cleaning or approximately half an hour for task model placement. The study will include approximately 12 participants.

What are the risks involved in this study?

Due to the use of distance, safety screens, or virtual environment there are no foreseeable risks to participating in this study.

What are the possible benefits of this study?

You will receive no direct benefit from participating in this study; however, this research will contribute to more intuitive future robot control interfaces.

Do you have to participate?

No, your participation is voluntary. You may decide not to participate at all or, if you start the study, you may withdraw at any time. Withdrawal or refusing to participate will not affect your relationship with The University of Texas at Austin in any way.

If you would like to participate please sign this form (see below) and give it to the researcher, Andrew Sharp. You will receive a copy of this form.

Will there be any compensation?

You will not receive any type of payment participating in this study.

How will your privacy and confidentiality be protected if you participate in this research study?

Your privacy and the confidentiality of your data will be protected. No personally identifying information of any kind will be collected, and so there will be no way to link any data to individual participants.

Figure D.4: IRB consent form for TVF operator evaluation page one.

If it becomes necessary for the Institutional Review Board to review the study records, information that can be linked to you will be protected to the extent permitted by law. Your research records will not be released without your consent unless required by law or a court order. The data resulting from your participation may be made available to other researchers in the future for research purposes not detailed within this consent form. In these cases, the data will contain no identifying information that could associate it with you, or with your participation in any study.

Whom to contact with questions about the study?

Prior, during, or after your participation you can contact the researcher Andrew Sharp at 660-342-3339 or send an email to asharp@utexas.edu for any questions or if you feel that you have been harmed.

Whom to contact with questions concerning your rights as a research participant?

For questions about your rights or any dissatisfaction with any part of this study, you can contact, anonymously if you wish, the Institutional Review Board by phone at (512) 471-8871 or email at orssc@uts.cc.utexas.edu.

Participation

If you agree to participate please sign this form and provide verbal consent.

Signature

You have been informed about this study's purpose, procedures, possible benefits and risks, and you have received a copy of this form. You have been given the opportunity to ask questions before you sign, and you have been told that you can ask other questions at any time. You voluntarily agree to participate in this study. By signing this form, you are not waiving any of your legal rights.

Printed Name

Signature

Date

As a representative of this study, I have explained the purpose, procedures, benefits, and the risks involved in this research study.

Print Name of Person obtaining consent

Signature of Person obtaining consent

Date

Figure D.5: IRB consent form for TVF operator evaluation page two.



OFFICE OF RESEARCH SUPPORT & COMPLIANCE

THE UNIVERSITY OF TEXAS AT AUSTIN

P.O. Box 7426, Austin, Texas 78713 · Mail Code A3200
(512) 471-8871 · FAX (512) 471-8873

FWA # 00002030

Date: 08/27/2018
PI: Andrew P Sharp
Dept: Engineering, Mechanical
Title: Verification of Task Virtual Fixtures Through Operator Evaluation

Re: IRB Expedited Approval for Protocol Number 2018-06-0092

Dear Andrew P Sharp,

In accordance with the Federal Regulations the Institutional Review Board (IRB) reviewed the above referenced research study and found it met the requirements for approval under the Expedited category noted below for the following period of time: 08/27/2018 to 08/26/2019, Expires 12 a.m. [midnight] of this date. If the research will be conducted at more than one site, you may initiate research at any site from which you have a letter granting you permission to conduct the research. You should retain a copy of the letter in your files.

- ☐ 1) Clinical studies of drugs and medical devices only when condition (a) or (b) is met. (a) Research on drugs for which an investigational new drug application (21 CFR Part 312) is not required. (Note: Research on marketed drugs that significantly increases the risks or decreases the acceptability of the risks associated with the use of the product is not eligible for expedited review). (b) Research on medical devices for which (i) an investigational device exemption application (21 CFR Part 812) is not required; or (ii) the medical device is cleared/approved for marketing and the medical device is being used in accordance with its cleared/approved labeling.
- ☐ 2) Collection of blood samples by finger stick, heel stick, ear stick, or venipuncture as follows: (a) from healthy, non-pregnant adults who weigh at least 110 pounds. For these subjects, the amounts drawn may not exceed 550 ml in an 8 week period and collection may not occur more frequently than 2 times per week; or (b) from other adults and children, considering the age, weight, and health of the subjects, the collection procedure, the amount of blood to be collected, and the frequency with which it will be collected. For these subjects, the amount drawn may not exceed the lesser of 50 ml or 3 ml per kg in an 8 week period and collection may not occur more frequently than 2 times per week.

Figure D.6: IRB approval letter for TVF operator evaluation page one.

- ☐ 3) Prospective collection of biological specimens for research purposes by non-invasive means.
Examples:
- (a) Hair and nail clippings in a non-disfiguring manner.
 - (b) Deciduous teeth at time of exfoliation or if routine patient care indicates a need for extraction;
 - (c) Permanent teeth if routine patient care indicates a need for extraction.
 - (d) Excreta and external secretions (including sweat).
 - (e) Uncannulated saliva collected either in an un-stimulated fashion or stimulated by chewing gumbase or wax or by applying a dilute citric solution to the tongue.
 - (f) Placenta removed at delivery.
 - (g) Amniotic fluid obtained at the time of rupture of the membrane prior to or during labor.
 - (h) Supra- and subgingival dental plaque and calculus, provided the collection procedure is not more invasive than routine prophylactic scaling of the teeth and the process is accomplished in accordance with accepted prophylactic techniques.
 - (i) Mucosal and skin cells collected by buccal scraping or swab, skin swab, or mouth washings.
 - (j) Sputum collected after saline mist nebulization.
- ☐ 4) Collection of data through non-invasive procedures (not involving general anesthesia or sedation) routinely employed in clinical practice, excluding procedures involving x-rays or microwaves. Where medical devices are employed, they must be cleared/approved for marketing. (Studies intended to evaluate the safety and effectiveness of the medical device are not generally eligible for expedited review, including studies of cleared medical devices for new indications).
Examples:
- (a) Physical sensors that are applied either to the surface of the body or at a distance and do not involve input of significant amounts of energy into the subject or an invasion of the subject's privacy.
 - (b) Weighing or testing sensory acuity.
 - (c) Magnetic resonance imaging.
 - (d) Electrocardiography, electroencephalography, thermography, detection of naturally occurring radioactivity, electroretinography, ultrasound, diagnostic infrared imaging, doppler blood flow, and echocardiography.
 - (e) Moderate exercise, muscular strength testing, body composition assessment, and flexibility testing where appropriate given the age, weight, and health of the individual.
- ☐ 5) Research involving materials (data, documents, records, or specimens) that have been collected, or will be collected solely for non-research purposes (such as medical treatment or diagnosis).
Note: Some research in this category may be exempt from the HHS regulations for the protection of human subjects. 45 CFR 46.101(b)(4). This listing refers only to research that is not exempt.
- ☒ 6) Collection of data from voice, video, digital, or image recordings made for research purposes.
- ☒ 7) Research on individual or group characteristics or behavior (including, but not limited to, research on perception, cognition, motivation, identity, language, communication, cultural beliefs or practices, and social behavior) or research employing survey, interview, oral history, focus group, program evaluation, human factors evaluation, or quality assurance methodologies.
Note: Some research in this category may be exempt from the HHS regulations for the protection of human subjects. 45 CFR 46.101(b)(2) and (b)(3). This listing refers only to research that is not exempt.
- ☒ Use the attached approved informed consent document(s).
- ☐ You have been granted a Waiver of Documentation of Consent according to 45 CFR 46.117 and/or 21 CFR 56.109(c)(1).
- ☐ You have been granted a Waiver of Informed Consent according to 45 CFR 46.116(d).

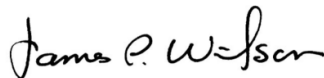
Figure D.7: IRB approval letter for TVF operator evaluation page two.

Responsibilities of the Principal Investigator:

1. Report immediately to the IRB any unanticipated problems.
2. Submit for review and approval by the IRB all modifications to the protocol or consent form(s). Ensure the proposed changes in the approved research are not applied without prior IRB review and approval, except when necessary to eliminate apparent immediate hazards to the subject. Changes in approved research implemented without IRB review and approval initiated to eliminate apparent immediate hazards to the subject must be promptly reported to the IRB, and will be reviewed under the unanticipated problems policy to determine whether the change was consistent with ensuring the subjects continued welfare.
3. Report any significant findings that become known in the course of the research that might affect the willingness of subjects to continue to participate.
4. Ensure that only persons formally approved by the IRB enroll subjects.
5. Use only a currently approved consent form, if applicable. Note: Approval periods are for 12 months or less.
6. Protect the confidentiality of all persons and personally identifiable data, and train your staff and collaborators on policies and procedures for ensuring the privacy and confidentiality of subjects and their information.
7. Submit a Continuing Review Application for continuing review by the IRB. Federal regulations require IRB review of on-going projects no less than once a year a reminder letter will be sent to you two months before your expiration date. If a reminder is not received from Office of Research Support and Compliance (RSC) about your upcoming continuing review, it is still the primary responsibility of the Principal Investigator not to conduct research activities on or after the expiration date. The Continuing Review Application must be submitted, reviewed and approved, before the expiration date.
8. Upon completion of the research study, a Closure Report must be submitted to the RSC.
9. Include the IRB study number on all future correspondence relating to this protocol.

If you have any questions contact the RSC by phone at (512) 471-8871 or via e-mail at orsc@uts.cc.utexas.edu.

Sincerely,



James Wilson, Ph.D.
Institutional Review Board Chair

Figure D.8: IRB approval letter for TVF operator evaluation page three.

Appendix E

Spatially Discrete Operator Evaluation Documents

Please read and fill out this consent form. Let me know if you have any questions.

The interface you are looking at is RViz. It is a robot visualization environment paired with Robot Operating System. These are open source packages upon which my research is built. My research is on Virtual Fixtures or VFs which are software implemented constraints similar to those in modeling assemblies or jigs in machining. Interactive markers are used to place the surface within the virtual environment (demonstrate on z4xy4 super ellipsoid model). You can also move the robot using an interactive marker (demonstrate). The goal of these tests is to have the highest percentage of the task surface points reachable by the robot. There are two versions of this task. In the first version of the task you will only be provided with the reachable percentage and in the second version you will be provided with reachability information on specific locations in the Virtual Fixture. For each trial I will be recording time, percentage of surface reached, a screenshot of your results, and any comments you would like recorded. You may now have up to five minutes to explore the environment if you would like.

To begin with please rate your experience with robotic manipulators from one to ten.

Now we will begin with the first super ellipsoid model. Please place it at a location where you think the robot will have high reachability and let me know when you are finished.

Your reachability percentage was BLANK. Please select a location where you think the robot will have a higher reachability and let me know when you're finished.

Your reachability percentage was BLANK. Please fill out this survey about your last test.

I will now let you examine the virtual fixture and select a location based on the reachable poses where you think the robot will have a higher level of reachability.

Your reachability percentage was BLANK. I will now let you examine the virtual fixture and select a location based on the reachable poses where you think the robot will have a higher level of reachability.

Please fill out this survey about your last test while I set up the next model.

Next, we will be testing a model pulled from an online database. The goal of this task is the same as the previous trials. Please place it at a location where you think the robot will have high reachability and let me know when you are finished.

Your reachability percentage was BLANK. Please select a location where you think the robot will have a higher reachability and let me know when you're finished.

Your reachability percentage was BLANK. Please fill out this survey about your last test.

Figure E.1: Script for spatially discrete task MTTT operator evaluation page one.

I will now let you examine the virtual fixture and select a location based on the reachable poses where you think the robot will have a higher level of reachability.

Your reachability percentage was BLANK. I will now let you examine the virtual fixture and select a location based on the reachable poses where you think the robot will have a higher level of reachability.

You have successfully completed this training. Please fill out this survey about your last test.

Would you like to continue searching for a location with higher reachability?

Do you have any additional comments about individual trials or the overall testing?

Figure E.2: Script for spatially discrete task MTTT operator evaluation page two.

Appendix F

Results from Spatially Discrete TVF Evaluation

Table F.1: User Results from Spatially Discrete TVF Evaluation

	Model	$N_{xy} = 0, N_z = 0$					$N_{xy} = 4, N_z = 0$				
User 1	Order	1					3				
Experience 4	Time	01:10	01:47	00:22	01:47		01:11	00:44	00:38	00:25	
Exploration 3:05	Percentage	68	68	50	66		66	71	79	74	
User 2	Order	3					1				
Experience 1	Time	00:34	01:15	00:42	00:46		02:43	00:44	01:19	00:38	
Exploration 5:00	Percentage	42	60	68	66		11	42	53	68	
User 3	Order	3					1				
Experience 8	Time	01:12	00:49	00:33	00:21		00:27	01:14	00:31	00:35	
Exploration 1:52	Percentage	58	58	72	62		50	26	84	58	
User 4	Order	1					2				
Experience 2	Time	00:24	01:21	00:53	00:05		00:47	00:02	01:01	00:22	
Exploration 1:32	Percentage	34	20	18	58		74	74	79	82	
User 5	Order	1					3				
Experience 3	Time	00:36	01:31	01:41	01:38		00:38	00:18	00:29	01:00	
Exploration 0:04	Percentage	50	60	56	56		66	63	71	71	
User 6	Order	2					1				
Experience 8	Time	03:55	03:16	03:48	02:25		00:05	01:15	01:23	01:09	
Exploration 0:15	Percentage	34	46	16	22		18	55	76	68	
User 7	Order	1					3				
Experience 7	Time	00:33	01:09	01:54	00:52		00:46	00:03	00:54	01:12	
Exploration 5:00	Percentage	30	70	54	62		84	61	61	74	
User 8	Order	1					2				
Experience 1	Time	02:05	02:12	03:35	04:02		01:41	02:03	03:09	04:41	
Exploration 5:00	Percentage	30	14	10	8		36	36	31	49	
User 9	Order	2					3				
Experience 6	Time	00:45	01:05	01:07	00:25		01:14	00:35	00:32	00:05	
Exploration 1:28	Percentage	44	54	56	60		53	63	63	74	
User 10	Order	3					2				
Experience 5	Time	01:32	01:17	01:02	01:16		02:01	00:26	01:27	01:02	
Exploration 3:19	Percentage	60	64	64	62		71	53	76	68	
User 11	Order	3					2				
Experience 6	Time	00:20	00:21	00:27	00:58		00:26	00:25	00:23	00:16	
Exploration 0:00	Percentage	42	44	58	78		45	66	66	71	

Table F.2: User Results from Spatially Discrete TVF Evaluation

	Model	$N_{xy} = 0, N_z = 4$				Maoi (908062)			
User 1	Order	2				4			
Experience 4	Time	01:51	02:01	01:02	01:23	01:18	01:07	00:34	00:57
Exploration 3:05	Percentage	92	21	85	85	23	66	74	84
User 2	Order	2				4			
Experience 1	Time	01:41	00:43	00:46	00:39	00:41	00:23	00:34	00:43
Exploration 5:00	Percentage	51	77	62	74	49	56	59	67
User 3	Order	2				4			
Experience 8	Time	01:01	00:25	00:15	00:45	02:17	01:04	00:35	00:43
Exploration 1:52	Percentage	67	90	82	85	23	26	52	59
User 4	Order	3				4			
Experience 2	Time	00:36	00:35	01:47	00:04	00:36	01:19	00:27	01:08
Exploration 1:32	Percentage	15	38	26	36	8	5	41	0
User 5	Order	2				4			
Experience 3	Time	01:17	00:02	02:08	00:32	01:49	00:53	01:34	00:52
Exploration 0:04	Percentage	92	97	95	89	51	20	0	56
User 6	Order	3				4			
Experience 8	Time	03:37	01:45	02:13	02:25	02:54	02:00	01:55	01:43
Exploration 0:15	Percentage	54	72	69	85	21	62	67	44
User 7	Order	2				4			
Experience 7	Time	01:47	01:17	01:13	01:02	01:46	00:04	00:25	01:37
Exploration 5:00	Percentage	72	51	72	69	15	72	90	87
User 8	Order	3				4			
Experience 1	Time	01:35	00:57	01:54	01:34	02:38	01:33	02:03	02:13
Exploration 5:00	Percentage	11	18	53	45	5	5	10	0
User 9	Order	1				4			
Experience 6	Time	01:28	00:05	01:32	00:33	01:12	00:52	00:29	01:07
Exploration 1:28	Percentage	49	62	44	69	31	0	28	18
User 10	Order	1				4			
Experience 5	Time	02:54	00:54	01:51	00:32	01:03	00:59	02:11	02:53
Exploration 3:19	Percentage	26	62	72	49	13	15	8	3
User 11	Order	1				4			
Experience 6	Time	01:00	00:02	00:45	00:31	00:14	00:35	00:25	00:26
Exploration 0:00	Percentage	51	64	56	36	15	26	36	33
									38

Table F.3: Average Results from Spatially Discrete TVF Evaluation

	Model	$N_{xy} = 0, N_z = 0$				$N_{xy} = 4, N_z = 0$			
TVF Poses	Avg Time	00:44	01:10	01:11	00:48	01:10	00:35	00:45	00:42
	Avg Percentage	45	60	61	61	53	51	66	69
No TVF Poses	Avg Time	01:34	01:42	01:41	01:46	01:02	00:49	01:20	01:19
	Avg Percentage	45	43	36	49	52	59	68	69

Table F.4: Average Results from Spatially Discrete TVF Evaluation

	Model	$N_{xy} = 0, N_z = 4$				Maoi (908062)			
TVF Poses	Avg Time	01:27	00:30	01:11	00:42	01:33	00:39	00:43	01:00
	Avg Percentage	66	75	71	77	34	35	46	57
No TVF Poses	Avg Time	01:56	01:02	01:35	01:05	01:27	01:15	01:16	01:33
	Avg Percentage	42	46	60	56	14	30	39	27
									55

Table F.5: Average Per Trial Results from Spatially Discrete TVF Evaluation

	Trial	1	2	3	4
TVF Poses	Avg Time	01:30	01:12	01:28	01:26
	Avg Percentage	38	44	51	50
No TVF Poses	Avg Time	01:13	00:44	00:58	00:48
	Avg Percentage	49	55	61	66

Table F.6: Average Results from Spatially Discrete TVF Evaluation

Model	$N_{xy}, N_z = 0, 0$		$N_{xy}, N_z = 4, 0$		$N_{xy}, N_z = 0, 4$		908062	Avg
TVF								
Poses Survey Avg	1	2	1	2	1	2		
This was a difficult task.	3.25	3.20	2.40	3.00	2.80	2.60	4.20	3.60
This was a frustrating task.	2.75	2.40	2.20	2.25	2.20	2.00	2.80	2.40
The interface was easy to use.	3.25	3.80	4.00	3.75	3.80	4.00	3.20	3.60
I successfully completed the task.	2.25	2.80	3.40	3.25	3.80	3.80	2.20	3.40
I would improve with practice.	4.50	4.20	4.60	3.75	4.20	4.00	4.40	4.28
I was unsure where to place the part.	4.00	3.40	3.00	3.00	2.80	3.00	3.80	3.31
No TVF								
Poses Survey Avg	1	2	1	2	1	2	1	2
This was a difficult task.	3.67	3.83	3.33	3.33	3.50	3.67	4.00	3.83
This was a frustrating task.	2.83	3.17	3.00	2.67	2.50	3.17	3.67	3.33
The interface was easy to use.	3.50	3.83	3.50	3.83	3.67	3.67	3.50	3.83
I successfully completed the task.	3.00	2.83	2.83	3.17	3.33	3.17	2.83	2.50
I would improve with practice.	4.17	4.17	4.17	4.17	4.33	4.17	3.50	3.50
I was unsure where to place the part.	3.83	3.50	3.17	2.83	3.33	3.50	3.83	3.48

Appendix G

Spatially Continuous Operator Evaluation Documents

ABB testing script

A new method for Task Virtual Fixture (TVF) generation from task surface models has been developed. Virtual fixtures are virtual constraints on robotic motion much like a physical jig. You have been selected for this testing due to previous ABB training and experience with ABB hardware. Your goal in this testing is to clean the task surface with a plasma pen attached to a 6 DOF ABB IRB 140. The task parameters are a speed of V2, a z0 zone, and distance to the surface of 0.5 cm. Post-test pixel evaluation of the surface will determine the portions under-clean, clean, and over-clean.

There are two methods available to complete this task. The first method is pose teaching as is demonstrated at ABB training. The second method provides TVF poses integrated with ABB's RobotStudio. RobotStudio will also be available during method one testing. Testing will start from either a hardware or RobotStudio backup template. Which test is performed first will be randomly assigned and program development will take place with a test surface made from a softer material the pieces used to perform trials.

The testing measurements collected will consist of time measurements (setup time, run time), success of the task (surface percentage cleaned), number of collisions, a questionnaire of Likert scale questions after each test. During testing participants will also be able to provide their comments on the process to the PI. Since none of you have had RobotStudio training, Brian O'Neil will now provide a short overview of the program.

- Apply changes
- Synchronize to Station / Synchronize to RAPID

Now that Brian has provided an overview, here are the specific steps to move your code from the simulation environment to the hardware system.

- Open FromBackup_template
- Create path in home tab
 - Shift-left-click to select poses
 - Right-click
 - Add to path
 - Choose rTVFPath and pose location in path
- Synchronize to RAPID
- Update speed (v2) and zone (z0) in RAPID
- Simulate execution
- Copy rTVFPath to hardware controller
- Apply changes
- Verify speed (V2) and zone (z0) in RAPID
- Release Write Access
- Test

Figure G.1: Script for spatially continuous task operator evaluation with ABB software and hardware.

Appendix H

Results from Spatially Continuous TVF Evaluation

Table H.1: Results from Spatially Continuous TVF Evaluation

	User 1			User 2			User 3			User 4		
	M	L	H	M	L	H	M	L	H	M	L	H
Session	2	1	3	1	2	3	2	3	1	1	3	2
Setup Time	30:41	18:32	36:00	2:27:28	25:35	30:18	09:14	00:00	33:32	56:20	11:03	35:15
Execution Time	03:48	09:41	22:05	11:28	09:43	09:03	01:52	00:00	19:18	14:45	09:32	16:25
Over clean	0.52	0.55	0.20	0.40	0.11	0.87	0.36	-	0.76	1.92	0.40	0.94
Clean	18.87	78.26	95.35	65.19	57.19	73.39	40.69	-	95.65	94.03	83.69	95.70
Unclean	80.61	21.19	4.44	34.41	42.70	25.74	58.95	-	3.58	4.06	15.91	3.36
Percentage per minute	0.60	4.23	2.65	0.44	2.24	2.43	4.40	0.00	2.86	1.67	7.61	2.71
Collisions	2	0	0	10	0	0	0	0	0	0	0	0

Table H.2: Average Results from Spatially Continuous TVF Evaluation

	Average		
	M	L	H
Manual (M), Low (L), High (H)			
Setup Time	01:00:56	13:47	33:46
Execution Time	07:58	07:14	16:43
Over clean	0.80	0.35	0.69
Clean	54.69	73.05	90.02
Unclean	44.51	26.60	9.28
Percentage per minute	1.78	3.52	2.66

Table H.3: Likert Survey Results from Spatially Continuous TVF Evaluation

	User 1			User 2			User 3			User 4			Average		
	M	L	H	M	L	H	M	L	H	M	L	H	M	L	H
Manual (M), Low (L), High (H)															
The interface was easy to use	4	4	4	5	5	4	2	-	3	5	4	5	4.00	4.33	4.00
This was a frustrating task	5	3	3	4	3	3	5	-	4	4	2	2	4.50	2.67	3.00
This was a difficult task	4	3	3	5	2	2	4	-	2	4	2	1	4.25	2.33	2.00
I successfully completed the task	1	4	5	2	4	1	3	-	5	4	3	5	2.50	3.67	4.00
Manually assigning poses was the most time consuming portion of the task	5	-	-	5	-	-	5	-	5	5	-	0	5.00	-	2.50
Having TVF poses available increased setup speed	5	5	5	3	5	4	5	-	4	-	5	5	4.33	5.00	4.50
Having TVF poses available increased process success	5	-	5	3	4	4	5	-	4	-	3	5	4.33	3.50	4.50
I would prefer to have TVF poses available for task setup in the future	5	5	5	3	5	5	5	-	4	-	5	5	4.33	5.00	4.75

Index

Abstract, vi

Acknowledgments, v

Bibliography, 223

Dedication, iv

Bibliography

- ABB. IRB 140. "<https://new.abb.com/products/robotics/industrial-robots/irb-140>", 2018a. Accessed: 2018-12-30.
- ABB. RobotStudio. "<https://new.abb.com/products/robotics/robotstudio>", 2018b. Accessed: 2018-12-16.
- ABB. Robtarget Documentation. "<http://developercenter.robotstudio.com/BlobProxy/manuals/RapidIFDTechRefManual/doc545.html>", 2018c. Accessed: 2018-12-17.
- J. J. Abbott, P. Marayong, and A. M. Okamura. Haptic Virtual Fixtures for Robot-Assisted Manipulation. In *Results of the 12th International Symposium ISRR*, pages 49–64, 2007.
- N. Aspert, T. Ebrahimi, and P. Vanderghenst. Non-linear subdivision using local spherical coordinates. *Computer Aided Geometric Design*, 20(EPFL-ARTICLE-86961):165–187, 2003.
- ASUS. *Xtion PRO LIVE Specifications*. ASUS, 2015. "https://www.asus.com/us/Multimedia/Xtion_PRO_LIVE/specifications/".
- Boost. The Boost Graph Library (BGL). "http://www.boost.org/doc/libs/1_66_0/libs/graph/doc/index.html", 2017. Accessed: 2017-12-26.

- M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, and B. Levy. *Polygon mesh processing*. AK Peters/CRC Press, 2010.
- S. A. Bowyer, B. L. Davies, and F. Rodriguez y Baena. Active constraints/virtual fixtures: A survey. *Robotics, IEEE Transactions on*, 30(1):138–157, 2014.
- D. Bruemmer, J. Marble, D. Dudenhoeffer, M. Anderson, and M. McKay. Intelligent Robots for Use in Hazardous DOE Environments. In *Performance Metrics for Intelligent Systems (PERMIS), Gaithersburg, MD*. Performance Metrics for Intelligent Systems (PERMIS), 2002.
- J. S. Byrd. ARIES: A Mobile Robot Inspector. In *American Nuclear Society Meeting on Robotics and Remote Systems*. American Nuclear Society, 1995.
- J. S. Byrd. An Intelligent Inspection and Survey Robot. In *Conference on Industry Partnerships to Deploy Environmental Technology*, 1996.
- J. S. Byrd and R. O. Pettus. A Robotic Inspector for Low-Level Radioactive Waste. In *C2nd Conference and Exposition Demonstration on Robotics for Challenging Environments*, 1996.
- R. A. Castillo-Cruces and J. Wahrburg. Virtual fixtures with autonomous error compensation for human–robot cooperative tasks. *Robotica*, 28:267–277, 2011.
- M. Chiou, N. Hawes, R. Stolkin, K. L. Shapiro, J. R. Kerlin, and A. Clouter. Towards the principled study of variable autonomy in mobile robots. In *2015 IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, 2015.

- M. Cindy Passmore, M. Alison E. Dobbie, M. Michael Parchman, and P. James Tysinger. Guidelines for Constructing a Survey. *Family Medicine*, 34:281–286, 2002.
- M. Ciocarlie, K. Hsiao, A. Leeper, and D. Gossow. Mobile Manipulation Through An Assistive Home Robot. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2012.
- B. P. DeJong, E. L. Faulring, J. E. Colgate, M. A. Peshkin, H. Kang, Y. S. Park, and T. F. Ewing. Lessons Learned From a Novel Teleoperation Testbed. *Industrial Robot: An International Journal*, 2006.
- J. Denavit and R. Hartenberg. *Kinematic Synthesis of Linkages*. McGraw-Hill, 1964.
- J. Denavit and R. S. Hartenberg. A Kinematic Notation For Lower-Pair Mechanisms Based On Matrices. *Journal of Applied Mechanics: ASME DC*, 1955.
- DOE. EM Update, Vol. 8, Issue 16, Aug. 31, 2016. "<https://content.govdelivery.com/accounts/USD0E0EM/bulletins/15f7c4b>", 2016. Accessed: 2016-08-31.
- S. Edwards. The Descartes Planning Library for Semi-Constrained Cartesian Trajectories. "https://roscon.ros.org/2015/presentations/ROSCon_Descartes.pdf", 2015a. Accessed: 2018-09-13.
- S. Edwards. The descartes planning library for semi-constrained cartesian trajectories. https://roscon.ros.org/2015/presentations/ROSCon_Descartes.pdf, Oct 2015b.
- S. Edwards and C. Lewis. ROS--Industrial--Applying the Robot Operating System (ROS) to Industrial Applications. In *IEEE Int. Conference on Robotics and Automation, ECHORD Workshop*, 2012.

- FARO. Cobalt. "<http://cobalt.faro.com/>", 2017. Accessed: 2017-12-18.
- FARO. Cobalt. "<https://www.faro.com/products/factory-metrology/faro-cobalt-array-imager/>", 2018. Accessed: 2018-1-29.
- A. Foorginejad and K. Khalili. Umbrella curvature: a new curvature estimation method for point clouds. *Procedia Technology*, 12:347–352, 2014.
- GAMMA Group, UNC Chapel Hill. FCL: A Flexible Collision Library. "http://gamma.cs.unc.edu/FCL/fcl_docs/webpage/generated/index.html", 2015. Accessed: 2015-07-21.
- R. C. Goertz. *Master-slave Manipulator*. Argonne National Laboratory, 1949.
- D. Gossow, A. Leeper, D. Hershberger, and M. Ciocarlie. Interactive markers: 3-d user interfaces for ros applications [ros topics]. *IEEE Robotics & Automation Magazine*, 18(4): 14–15, 2011.
- D. Guan, L. Yan, Y. Yang, and W. Xu. A small climbing robot for the intelligent inspection of nuclear power plants. In *Information Science and Technology (ICIST), 2014 4th IEEE International Conference on*, pages 484–487, April 2014. doi: 10.1109/ICIST.2014.6920522.
- T. Harden and P. Pittman. Development of a robotic system to clean out spherical dynamic experiment containment vessels. In *American Nuclear Society EP&R and RR&S Topical Meeting*, pages 358–364, 2008.
- A. Hatna, R. Grieve, and P. Broomhead. Offsetting 3d contours on parametric surfaces. *The International Journal of Advanced Manufacturing Technology*, 16(3):189–195, 2000.

- K. P. Hawkins. Analytic Inverse Kinematics for the Universal Robots UR-5/UR-10 Arms. Technical report, Georgia Institute of Technology, 2013.
- F. Hazen. Drum inspection robots: Application development. In *Conference: Waste management '96*, 1996.
- P. Hebert, M. Bajracharya, J. Ma, N. Hudson, A. Aydemir, J. Reid, C. Bergh, J. Borders, M. Frost, M. Hagman, J. Leichty, P. Backes, B. Kennedy, P. Karplus, B. Satzinger, K. Byl, K. Shankar, and J. Burdick. Mobile Manipulation and Mobility as Manipulation-Design and Algorithms of RoboSimian. *Journal of Field Robotics*, 32:255–274, 2015.
- A. Hentout, M. R. Benbouali, B. B. I. Akli, and L. Melkou. A Telerobotic Human/Robot Interface for Mobile Manipulators: A Study of Human Operator Performance. In *IEEE International Conference on Control, Decision and Information Technologies (CoDIT)*. IEEE, 2013.
- L. M. Hiatt and R. Simmons. Coordinate Frames in Robotic Teleoperation. In *International Conference on Intelligent Robots and Systems*. IEEE, 2006.
- P. Hilton and A. Khan. New developments in laser cutting for nuclear decommissioning. *WM2014 Conference, March , 2014, Phoenix, AZ*, 4 2014.
- House of Commons Committee of Public Accounts. Uk parliament select committee report (2013): Nuclear decommissioning authority: Managing risks at sellafeld ltd. "<https://publications.parliament.uk/pa/cm201213/cmselect/cmpubacc/746/746.pdf>", 2013. Accessed: 2018-01-09.

- Hsi-Yung Feng and Huiwen Li. Constant scallop-height tool path generation for three-axis sculptured surface machining. *Computer-Aided Design*, 34(9):647–654, 2002.
- Innovmetric. PolyWorks Inspector. "<https://www.innovmetric.com/en/products/polyworks-inspector>", 2018a. Accessed: 2018-02-01.
- Innovmetric. PolyWorks Modeler. "<https://www.innovmetric.com/en/products/polyworks-modeler>", 2018b. Accessed: 2018-02-09.
- Innovmetric. PolyWorks. "<https://www.innovmetric.com/en>", 2018c. Accessed: 2018-03-09.
- A. Khan and P. Hilton. Fibre delivered laser beams-an alternative cost effective decommissioning technology. http://www.ocrobotics.com/downloads/Website/LS2/LaserSnake2%20TWI%20-%20NOLAMP%2014_Ali%20Khan_TWI.pdf, 9 2013.
- S. N. Kosari, F. Rydén, T. S. Lendvay, B. Hannaford, and H. J. Chizeck. Forbidden region virtual fixtures from streaming point clouds. *Advanced Robotics*, 28(22):1507–1518, 2014.
- K. Kruusamae and M. Pryor. High-precision telerobot with human-centered variable perspective and scalable gestural interface. In *2016 9th International Conference on Human System Interactions (HSI)*, pages 190–196. IEEE, 2016.
- S. N. Laboratories. ‘Mighty Mouse’ robot frees stuck radiation source. "<https://share.sandia.gov/news/resources/releases/2005/manuf-tech-robotics/mm-robot.html>", 2005. Accessed: 2015-09-30.

- M. Li, M. Ishii, and R. Taylor. Spatial motion constraints using virtual fixtures generated by anatomy. In *ROBOTICS, IEEE TRANSACTIONS ON*, volume 23. IEEE, 2007.
- J. Ma, H.-Y. Feng, and L. Wang. Normal vector estimation for point clouds via local delaunay triangle mesh matching. *Computer-Aided Design and Applications*, 10(3):399–411, 2013.
- J. Ma, J. Luo, H. Pu, Y. Peng, S. Xie, and J. Gu. Design, simulation and manufacturing of a tracked robot for nuclear accidents. In *Robotics and Biomimetics (ROBIO), 2014 IEEE International Conference on*, pages 1828–1833, Dec 2014. doi: 10.1109/ROBIO.2014.7090601.
- Men in Black. bowl. "<https://3dwarehouse.sketchup.com/model/6dc5e034-c223-4b84-9c13-96511f451665/bowl>", 2018. Accessed: 2018-10-03.
- MeshLab. MeshLab. "<http://www.meshlab.net/>", 2018. Accessed: 2018-10-03.
- K. Nagatani, S. Kiribayashi, Y. Okada, Otake, K. Yoshida, S. Tadokoro, and S. Kawatsuma. Emergency Response to the Nuclear Accident at the Fukushima Daiichi Nuclear Power Plants Using Mobile Rescue Robots. *Field Robotics*, 30:44–63, 2013.
- S. Nia Kosari, F. Rydén, T. S. Lendvay, B. Hannaford, and H. J. Chizeck. Forbidden region virtual fixtures from streaming point clouds. *Advanced Robotics*, 28(22):1507–1518, 2014.
- OGRE. Torus Knot Software. "<http://www.ogre3d.org/>", 2015. Accessed: 2015-07-21.
- Open Source Robotics Foundation. Display Types. "<http://wiki.ros.org/rviz/DisplayTypes>", 2015. Accessed: 2015-07-21.
- OpenMP. OpenMP. "<https://www.openmp.org/>", 2018. Accessed: 2018-10-01.

- J. Pan, S. Chitta, and D. Manocha. FCL: A General Purpose Library for Collision and Proximity Queries. *IEEE International Conference on Robotics and Automation*, may 2012.
- Y. S. Park, X. Zhao, and S. Korthals. Simulation of augmented telerobotic operation. In *Optomechatronic Technologies (ISOT), 2014 International Symposium on*, pages 242–246. IEEE, 2014.
- PCL. The CloudViewer. "http://pointclouds.org/documentation/tutorials/cloud_viewer.php#cloud-viewer", 2018. Accessed: 2018-10-4.
- I. Photonics. FLC 30 Cutting Head Brochure. <http://www.ipgphotonics.com/en/162/Widget/FLC+30+Cutting+Head+Brochure.pdf>, 2018. Accessed: 2018-01-10.
- M. Pryor. Operator Evaluation of Mobile Robotic Systems for Inventory, Inspection, and Contamination Survey Tasks in D&D Environments. *WM2017 Conference, March , 2017, Phoenix, AZ*, March 2017.
- M. Pryor and S. Landsberger. Mobile manipulation and survey system for H-Canyon and other applications across the DOE complex. *WM2017 Conference, March , 2017, Phoenix, AZ*, March 2017.
- PTC. CREO. "<http://www.ptc.com/cad/creo>", 2017. Accessed: 2017-03-02.
- PVATePla. PlasmaPen System. "<http://www.pvateplaamerica.com/pen.php>", 2018. Accessed: 2018-12-30.

- M. Quigley, K. Conley, B. P. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng. ROS: an open-source Robot Operating System. In *ICRA Workshop on Open Source Software*, 2009.
- M. Quigley, B. Gerkey, and W. D. Smart. *Programming Robots with ROS A Practical Introduction to the Robot Operating System*. O'Reilly, 2015.
- M. S. Rahman. *Graphs and Their Applications*, pages 1–9. Springer International Publishing, Cham, 2017. ISBN 978-3-319-49475-3. doi: 10.1007/978-3-319-49475-3_1. URL https://doi.org/10.1007/978-3-319-49475-3_1.
- Ratnesh Madaan. Descartes Package Summary. <http://wiki.ros.org/descartes>", 2015. Accessed: 2015-07-17.
- J. Ren, R. Patel, K. McIsaac, G. Guiraudon, and T. Peters. Dynamic 3-d virtual fixtures for minimally invasive beating heart procedures. In *MEDICAL IMAGING, IEEE TRANSACTIONS ON*, volume 27. IEEE, 2008.
- ROS-Industrial. ROS-Industrial Description. "<http://rosindustrial.org/about/description/>", 2015a. Accessed: 2015-07-21.
- ROS-Industrial. ROS-Industrial Packages. "<https://github.com/ros-industrial>", 2015b. Accessed: 2015-07-21.
- ROS Industrial Consortium. Issue 204. "<https://github.com/ros-industrial-consortium/descartes/issues/204>", 2018. Accessed: 2018-12-27.

- L. Rosenberg. Virtual fixtures: Perceptual tools for telerobotic manipulation. In *Virtual Reality Annual International Symposium*. IEEE, 1993.
- R. B. Rusu. *Semantic 3D Object Maps for Everyday Manipulation in Human Living Environments*. PhD thesis, Computer Science department, Technische Universitaet Muenchen, Germany, 10 2009.
- R. B. Rusu and S. Cousins. 3D is here: Point Cloud Library (PCL). In *IEEE International Conference on Robotics and Automation (ICRA)*, Shanghai, China, 5 2011.
- R. B. Rusu and S. Cousins. PCL Normal Estimation. "http://pointclouds.org/documentation/tutorials/normal_estimation.php#normal-estimation", 2017a. Accessed: 2017-02-24.
- R. B. Rusu and S. Cousins. PCL Walkthrough. "<http://pointclouds.org/documentation/tutorials/walkthrough.php>", 2017b. Accessed: 2017-02-24.
- F. Rydén and H. J. Chizeck. Forbidden-region virtual fixtures from streaming point clouds: Remotely touching and protecting a beating heart. In *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, pages 3308–3313. IEEE, 2012.
- F. Rydén, H. J. Chizeck, S. N. Kosari, H. King, and B. Hannaford. Using kinect and a haptic interface for implementation of real-time virtual fixtures. In *Proceedings of the 2nd Workshop on RGB-D: Advanced Reasoning with Depth Cameras (in conjunction with RSS 2011)*. RSS, 2011.
- Saling, James and Fentiman, Audeen. *Radioactive Waste Management*. Taylor and Francis, 2001.

- S. Schaefer, E. Vouga, and R. Goldman. Nonlinear subdivision through nonlinear averaging. *Computer Aided Geometric Design*, 25(3):162–180, 2008.
- K. Schroeder and M. Pryor. Framework for use of generalized force and torque data in transitional levels of autonomy. In *IEEE International Conference on Intelligent Robotics and Applications*, Aachen, Germany, 12 2011.
- K. Schroeder, M. Pryor, and T. Harden. A Black Box Model for Estimating Joint Torque in an Industrial Serial Manipulator. In *ASME 2013 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Portland OR. ASME, 2013.
- Scott Niekum. ar_track_alvar. "https://github.com/sniekum/ar_track_alvar", 2016. Accessed: 2016-08-31.
- A. Sharp and M. Pryor. Operator Control Modes for Implementing Autonomous Search Behaviors. In *ANS Student Conference*. American Nuclear Society, 2015.
- A. Sharp and M. Pryor. Variable Normal Surface Virtual Fixtures (VNSVF) for Semi-autonomous Task Completion. In *ANS Decommissioning and Remote Systems (D&RS) Joint Topical Meeting*. American Nuclear Society, 2016.
- A. Sharp and M. Pryor. Data Driven Virtual Fixtures for Improved Shared Control. In *2018 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*. ASME, 2018.

- A. Sharp, M. Horn, and M. Pryor. Operator Training for Preferred Manipulator Trajectories in a Glovebox. In *2017 IEEE International Workshop on Advanced Robotics and its Social Impacts*. IEEE, 2017a.
- A. Sharp, K. Kruusamae, B. Ebersole, and M. Pryor. Semiautonomous Dual-Arm Mobile Manipulator System with Intuitive Supervisory User Interfaces. In *2017 IEEE International Workshop on Advanced Robotics and its Social Impacts*. IEEE, 2017b.
- A. Sharp, C. Petlowany, and M. Pryor. Virtual Fixture Augmentation of Operator Selection of Non-contact Material Reduction Task Paths. In *2018 International Conference on Nuclear Engineering*. ASME, 2018.
- B. Siciliano and O. Khatib. *Springer Handbook of Robotics*. Springer, 2008.
- SketchUp. SketchUp 3D Warehouse. "<https://3dwarehouse.sketchup.com/>", 2017a. Accessed: 2017-03-02.
- SketchUp. SketchUp. "<https://sketchup.com/>", 2017b. Accessed: 2017-03-02.
- SOLIDWORKS. SOLIDWORKS. "<http://www.solidworks.com/>", 2017. Accessed: 2017-12-26.
- stratasys. stratasys Vero. "<https://www.stratasys.com/materials/search/vero>", 2018. Accessed: 2018-12-30.
- I. Sucan, M. Moll, and L. Kavraki. The Open Motion Planning Library. *IEEE Robotics and Automation Magazine*, dec 2012.

- I. A. Sucan and S. Chitta. MoveIt! "<http://moveit.ros.org/>", 2017a. Accessed: 2017-02-14.
- I. A. Sucan and S. Chitta. MoveIt! Robots. "<http://moveit.ros.org/robots/>", 2017b. Accessed: 2017-02-14.
- The Open Motion Planning Library. OMPL Available Planners. "<http://ompl.kavrakilab.org/planners.html>", 2015. Accessed: 2015-07-21.
- The Orocos Project. KDL wiki. "<http://www.orocos.org/kdl>", 2015a. Accessed: 2015-07-21.
- The Orocos Project. Open Robot Control Software. "<http://www.orocos.org/>", 2015b. Accessed: 2015-07-21.
- The Visualization Toolkit. The Visualization Toolkit. "<https://www.vtk.org/>", 2018. Accessed: 2018-03-06.
- Thingiverse. Thingiverse. "<https://www.thingiverse.com/>", 2018. Accessed: 2018-09-12.
- C. J. Turner, T. A. Harden, and J. A. Lloyd. Robotics in nuclear materials processing at lanl: Capabilities and needs. In *ASME 2009 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, pages 701–710. American Society of Mechanical Engineers, 2009.
- United States Department of Energy. DOE mission. "<http://energy.gov/mission>", 2015. Accessed: 2015-07-17.

- United States Department of Health and Human Services. Code of Federal Regulations Title 45: Public Welfare. "https://www.ecfr.gov/cgi-bin/retrieveECFR?gp=&SID=83cd09e1c0f5c6937cd9d7513160fc3f&pitd=20180719&n=pt45.1.46&r=PART&ty=HTML#se45.1.46_111", 2018. Accessed: 2018-12-26.
- United States Nuclear Regulatory Commission. Code of Federal Regulations. "<https://www.nrc.gov/reading-rm/doc-collections/cfr/part020/part020-1003.html>", 2018. Accessed: 2018-01-12.
- L. K. Wood. Advanced Recover and Integrated Extraction System (ARIES) Preconceptual Design Report. Technical report, Los Alamos National Laboratory, sep 1996.
- Xiuzhi Qu and Brent Stucker. A 3D surface offset method for STL-format models. *Rapid Prototyping Journal*, 9(3):133–141, 2003.
- T. Yamamoto, N. Abolhassani, S. Jung, A. M. Okamura, and T. N. Judkins. Augmented reality and haptic interfaces for robot-assisted surgery. *The International Journal of Medical Robotics and Computer Assisted Surgery*, 8(1):45–56, 2012.
- H. A. Yanco, A. Norton, W. Ober, D. Shane, A. Skinner, and J. Vice. Analysis of human-robot interaction at the darpa robotics challenge trials. *Journal of Field Robotics*, 32(3):420–444, 2015.
- M. Zhihong, C. Guo, M. Yanzhao, and K. Lee. Curvature estimation for meshes based on vertex normal triangles. *Computer-Aided Design*, 43(12):1561–1566, 2011.

Vita

Andrew Sharp was born in Missouri in 1990 to Rick and Cynthia Sharp. He graduated from Missouri University of Science and Technology with a B.S. in Mechanical Engineering and a minor in Physics in 2013. He continued his education by attending grad school at The University of Texas at Austin as part of the Nuclear Robotics Group.

Email address: asharp@utexas.edu

This dissertation was typeset with \LaTeX^\dagger by Andrew Sharp.

[†] \LaTeX is a document preparation system developed by Leslie Lamport as a special version of Donald Knuth's \TeX Program.